

---

# Road-, Air- and Water-based Future Internet Experimentation

<b>Project Acronym:</b>	<b>RAWFIE</b>		
<b>Contract Number:</b>	<b>645220</b>		
<b>Starting date:</b>	<b>Jan 1st 2015</b>	<b>Ending date:</b>	<b>Dec 31st, 2018</b>

<b>Deliverable Number and Title</b>	<b>D4.6 - Pilot Experimentation Scenarios for Validation and Testing</b>		
<b>Confidentiality</b>	PU	<b>Deliverable type<sup>1</sup></b>	R
<b>Deliverable File</b>	<b>RAWFIE-D4.6-Pilot Experimentation Scenarios for Validation and Testing_v02.docx</b>	<b>Date</b>	30.09.2016
<b>Approval Status<sup>2</sup></b>	<b>WP leader, 1st Reviewer, 2nd Reviewer</b>	<b>Version</b>	1.0
<b>Contact Person</b>	<b>Ph. Dallemagne</b>	<b>Organization</b>	CSEM
<b>Phone</b>		<b>E-Mail</b>	Philippe.Dallemagne@csem.ch

---

<sup>1</sup> Deliverable type: P(Prototype), R (Report), O (Other)

<sup>2</sup> Approval Status: WP leader, 1<sup>st</sup> Reviewer, 2<sup>nd</sup> Reviewer, Advisory Board



**AUTHORS TABLE**

<b>Name</b>	<b>Company</b>	<b>E-Mail</b>
Ph. Dallemagne	CSEM	Philippe.Dallemagne@csem.ch
Giovanni Tusa	IES	g.tusa@iessolutions.eu
Nikolaos Priggouris	HAI	PRIGGOURIS.Nikolaos@haicorp.com
Kakia Panagidi	UoA	kakiap@di.uoa.gr
Kostantinos Kolomvatsos	UoA	kostasks@di.uoa.gr
Miquel Cantero	ROBOTNIK	mcantero@robotnik.es
Marcel Heckel	FRAUNHOFER	Marcel.Heckel@ivi.fraunhofer.de
Cveta Dimitrova	EPSILON	cveta.dimitrova@epsilon-bulgaria.com
Vasil Kumanov	EPSILON	vasil.kumanov@epsilon-bulgaria.com
Kiriakos Georgouleas	HAI	GEORGOULEAS.Kiriakos@haicorp.com
Savvas Chatzchristofis	CERTH	savvash@gmail.com
Ricardo Martins	MST Oceanscan	rasm@oceanscan-mst.com
Lionel Blondé	HESSO	lionel.blonde@hesge.ch

**REVIEWERS TABLE**

<b>Name</b>	<b>Company</b>	<b>E-Mail</b>
Kakia Panagidi	UoA	kakiap@di.uoa.gr
Savvas Chatzchristofis	CERTH	savvash@gmail.com

**DISTRIBUTION**

<b>Name / Role</b>	<b>Company</b>	<b>Level of confidentiality<sup>3</sup></b>	<b>Type of deliverable</b>
All		PU	R

<sup>3</sup> Deliverable Distribution: PU (Public, can be distributed to everyone), CO (Confidential, for use by consortium members only), RE (Restricted, available to a group specified by the Project Advisory Board).



### CHANGE HISTORY

<b>Version</b>	<b>Date</b>	<b>Reason for Change</b>	<b>Pages/Sections Affected</b>
0.1	8.09.2016	First draft, merging all inputs	N/A
0.2	22.09.2016	Second draft with major revision	All
0.3	26.09.2016	Third draft with major revision	All
0.5	29.09.2016	Fourth distributed draft	All
1.0	07.10.2016	First release of the second document iteration	All



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

### **Abstract:**

This deliverable is based on the results of T4.1 for what concerns the definition of the testing and validation scenarios of the RAWFIE platform. It contains the definition of a set of test and validation scenarios that will be performed in WP6 as well as the definition of the metrics and the success criteria.

### **Keywords:**

Tested platform component system verification validation tests scenarios end users metrics



## **Part II: Table of Contents**

Part II: Table of Contents.....	5
List of Figures .....	7
List of Tables.....	8
1.1 Recommendation processing.....	11
Part III: Executive Summary .....	13
Part IV: Main Section .....	14
2 Introduction .....	14
2.1 Scope of D4.6.....	14
2.2 Abbreviations .....	15
3 Object of the validation and testing.....	16
3.1 Verification.....	16
3.2 Validation and evaluation.....	17
3.3 RAWFIE federation lifecycle.....	17
3.4 Verification and validation infrastructure and procedures .....	18
3.4.1 Non regression and stress tests .....	18
3.4.2 EDL Testing.....	19
4 Stakeholders and actors .....	20
5 Metrics (IES) .....	20
5.1 Introduction .....	20
5.2 Metric template definition.....	22
5.3 Success criteria.....	22
5.4 Platform metrics .....	22
5.5 Testbed metrics .....	29
5.6 UxV metrics .....	31
5.7 Interconnectivity (aka. communication) metrics.....	31
6 Verification.....	33
6.1 Verification scenarios.....	34
6.1.1 Frontend Tier (Web Portal GUI elements) .....	34
6.1.2 Middle Tier (Services and Communication components) .....	54
6.1.3 Testbed Tier (Testbeds and Resources control components) .....	78
6.2 Integrated system testing.....	100
7 Validation scenarios .....	100



7.1	User defined scenarios .....	101
7.1.1	Monitoring of Water Canals .....	102
7.1.2	Border Surveillance or Perimeter protection of large areas.....	104
7.1.3	On demand deployable Internet facilities .....	105
7.1.4	Exploration & Assessment of Network Technologies Robustness.....	106
7.1.5	Efficient Coordination for phenomena or mission.....	107
7.1.6	Over the Air (OTA) UxV Re-programming .....	108
7.1.7	Additional scenarios from Open calls.....	108
7.1.8	Efficient coordination of multiple UxVs .....	115
7.2	RAWFIE Platform Admin scenarios.....	116
7.2.1	Administrator manages the user rights .....	116
7.2.2	Administrators adds a new user .....	116
7.2.3	System monitoring and error notifications .....	117
7.3	Testbed operator scenarios .....	117
7.3.1	Schedule maintenance of resources .....	117
7.3.2	Cancel running experiment .....	118
7.3.3	Connect a new Testbed to the RAWFIE platform .....	120
7.4	UxV Manufacturers scenarios .....	120
7.4.1	Install new UxVs in a testbed .....	120
7.4.2	Autonomous coordination of multiple UxVs.....	121
7.5	Early sub-system tests and validation .....	123
7.5.1	UxV Data Generator .....	124
7.5.2	UGV navigation and monitoring .....	126
8	ANNEX 1. Validation scenario template .....	127
9	ANNEX 2. Component testing - how to read the verification scenarios .....	129
10	ANNEX 3: Analysis of the first Open call .....	131
11	ANNEX: Unreferenced Requirements .....	135
12	References.....	140



## **List of Figures**

No table of figures entries found. (left for future release)



## **List of Tables**

Table 1: Abbreviations.....	15
Table 2: Metrics template .....	22
Table 3: Platform metrics.....	28
Table 4: Testbed metrics .....	29
Table 5: UxV metrics.....	31
Table 6: Communication metrics.....	33
Table 7: Verification test of the Web Portal - Login/ Logout.....	34
Table 8: Verification test of the Web Portal – Language selection .....	34
Table 9: Verification test of the Wiki Tool – Component Help .....	36
Table 10: Verification test of the Browse testbeds and UxVs and start booking .....	37
Table 11: Verification test of the Booking Tool Calendar View and its display options.....	38
Table 12: Verification test of the Booking Tool Calendar View Interactions.....	39
Table 13: Verification test of the Booking Tool Create Reservation .....	40
Table 14: Verification test of the Booking Tool Edit Reservation Actions.....	41
Table 15: Verification test of the in-Textual Editor Experiments definition.....	43
Table 16: Verification test of the Textual Editor Experiments Update .....	44
Table 17: Verification test of the in-Visual Editor Experiments Define .....	45
Table 18: Verification test of the in-Visual Editor Experiments Update.....	46
Table 19: Verification test of the Editor switching.....	46
Table 20: Verification test of the experiment Launchings.....	47
Table 21: Verification test of the Visualization of experiment status .....	48
Table 22: Verification test of the canceling of experiments .....	48
Table 23: Verification test of the Visualisation of system and UxV health status .....	49
Table 24: Verification test of the UxV navigation tool access and produced instructions validation.....	49
Table 25: Verification test of the User request handling .....	50
Table 26: Verification test of the Geospatial data handling .....	50
Table 27: Verification test of the Geospatial data modification .....	51
Table 28: Verification test of the Experiment Controller communication .....	51
Table 29: Verification test of the Visualization Tool Interaction .....	52
Table 30: Verification test of the Camera interaction.....	52
Table 31: Verification test of the provision of an interface to the Analysis Engine by the Analysis Tool.....	52
Table 32: Verification test of the ability of the Analysis Tool to query available data schemas .....	53
Table 33: Verification test of the ability of the Analysis Tool to read results from the results database.....	53
Table 34: Verification test of the resource Retrieval from testbed facility.....	54
Table 35: Verification test of the Addition of a new testbed facility to the RAWFIE federation .....	55
Table 36: Verification test of the Registration of a new UxV node into a testbed facility.....	56
Table 37: Verification test of the Retrieval of testbed information and belonging resources .....	57





Table 38: Verification test of the Experiments compilation .....58

Table 39: Verification test of the Experiments validation .....58

Table 40: Verification test of the Users & Rights Service login checking .....59

Table 41: Verification test of the Users & Rights Service roles/rights checking .....59

Table 42: Verification test of the user rights checks.....60

Table 43: Verification test of Booking Service add reservation functionality .....61

Table 44: Verification test of Booking Service edit reservation functionality .....62

Table 45: Verification test of Booking Service approve reservation functionality .....63

Table 46: Verification test of Booking Service reject reservation functionality .....64

Table 47: Verification test of Booking Service delete reservation functionality.....65

Table 48: Verification test of Booking Service retrieve reservation(s) functionality.....66

Table 49: Verification test of Booking Service check for conflicts functionality .....66

Table 50: Verification test of Booking Service simultaneous reservations support .....67

Table 51: Verification test of the Launching Service manualStart (short term launching) ....68

Table 52: Verification test of the Launching Service schedule (long term launching) .....69

Table 53: Verification test of the Launching Service cancellation request .....71

Table 54: Verification test of Launching Service simultaneous launching capability .....72

Table 55: Visualisation engine user request handling .....72

Table 56: Visualisation engine user request handling .....73

Table 57: Visualization engine geospatial data modification .....73

Table 58: Visualization engine camera interaction.....74

Table 59: Verification test of the ability of the Analysis Engine to query message bus streams & schemas from the schema registry .....74

Table 60: Verification test of the ability of the Analysis Engine to receive messages from the Analysis Tool.....74

Table 61: Verification test of the ability of the Analysis Engine to write data to the results database.....75

Table 62: Verification test of the System Monitoring .....76

Table 63: Verification test of the System Monitoring Problem Notifications.....76

Table 64: Verification test of the Accounting data collection .....77

Table 65: Verification test of Experiment Controller connection .....77

Table 66: Verification test of Experiment Controller workflow .....78

Table 67: Verification test of Monitoring Activity.....79

Table 68: Verification test of network interface switching due to connectivity problems.....79

Table 69: Verification test of Connection and of Accuracy validation of the given Instructions .....80

Table 70: Verification test of Proximity component Backup communication .....81

Table 71: Verification test of UxV retrieval using the communication system of the Proximity component.....82

Table 72: Verification test of Swarm motion using the Proximity component .....82

Table 73: Verification test of Testbed health status.....86

Table 74: Verification test of UxV Return to base .....87

Table 75: Verification test of the ability of the UxV to follow a route .....88

Table 76: Verification test of Acquire sensor samples .....89



Table 77: Verification test of Fidelity to commands .....90  
Table 78: Verification test of Continuous communication.....91  
Table 79: Verification test of Secure communication .....92  
Table 80: Verification test of Real-time communication .....93  
Table 81: Verification test of Resume communication and data transfer.....94  
Table 82: Verification test of UxV Device Management .....95  
Table 83: Verification test of the UxV connection.....96  
Table 84: Verification test of Sensor Data Acquisition 1 .....97  
Table 85: Verification test of Sensor Data Acquisition 2.....98  
Table 86: Verification test of Data Storage .....99  
Table 87: Verification test of Waypoints Processed.....100  
Table 88: Validation scenario: Scenario 1 .....127  
Table 89: specific validation scenario: xxxx .....129  
Table 90: Metrics and success criteria .....129  
Table 91: test for the component (or sub-component).....130  
Table 92: tests that will be performed on a given component .....130  
Table 93: specific test of a given component and the expected results .....131  
Table 94: Additional scenarios from Open calls.....131  
Table 95 – Unreferenced Requirements .....135

## Foreword

The first version of the deliverable “Pilot Experimentation Scenarios for Validation and Testing” (D4.3) introduced the plan and the approach that is followed to perform and document the tests for verification and validation of the RAWFIE system. The second iteration of this deliverable focuses on the needs of the stakeholders that will participate in the context of the Open Calls.

While retaining most of the previous work reflected in D4.3, D4.6 adds the description of the scenarios and uses cases that corresponds to the needs of newcomers and their uses of the RAWFIE system.

A third iteration of this document will later complete the process of identification and description of the scenarios for validation and testing.

### **Plans for addressing the reviewers’ recommendations after the first review**

Taking into account the iterative process adopted in the project, and therefore the fact that each deliverable type, and so the one reporting on “*Pilot Experimentation Scenarios for Validation and Testing*”, is submitted at regular intervals corresponding to the different cycles of requirements, design, verification and validation planning and implementation, in the next iterations of this deliverable the consortium will take the actions needed to follow the recommendations received after every review.



In the following, we explain how RAWFIE partners have addressed, or intend to address the above-mentioned recommendations in the subsequent versions of this deliverable D4.9 (M30).

The D4.3 document included the complete list of verification tests that were identified as relevant during the first cycle, at a very early stage of the project, to ensure an extensive component and system test campaign. After the first and second implementation rounds, some tests may prove unnecessary and should be deleted from subsequent versions of the document. The open call led to the selection of several proposals from various new RAWFIE stakeholders, which the consortium analysed and reported in D4.6. The analysis is done from several perspectives: the needs and requirements expressed by newcomers, the identified satisfaction levels and the corresponding metrics and the typical scenarios and use cases. D4.6 justifies the presence of scenarios and tests from the user and needs perspectives (in particular by tracing back to the requirements). In D4.9, the verification and validation tests described in Section 6 and Section 7, will be kept only if they relate to any specific requirement appearing respectively in D3.2 and then D3.3.

Updated or new requirements coming from WP3, will be in turn reflected in the functionalities described in the architecture and design deliverables (D4.4, D4.5, D4.7, D4.8). Tests related to functionalities that are not explicitly mentioned in those deliverables, will not be considered as well, or existing tests will be updated accordingly.

It should also be noted that, with the preparation of deliverable D6.1, the consortium took the opportunity to proceed with the update or the removal of all tests that were not applicable anymore, after the first implementation cycle was completed.

As recommended and already stated in the first release of this deliverable (D4.3) on page 21, the consortium defines in D4.6 the success criteria for the evaluation of the platform, and refines them in deliverables D4.9.

### **1.1 Recommendation processing**

This section describes how the recommendations given by the reviewers during the first review meeting in Porto are addressed and implemented.

- (1) The deliverable was delivered on time.
- (2) The report does not appear to have been reviewed, or the review is not reported in the version list.

The list of internal reviewers was mentioned. The original reviewers have been retained for the present document. The reviewing process will be announced internally and the results will be considered for improvement of the document.

- (3) The traceability to requirements is weak.

Requirements linked to a given scenario are mentioned in its description. The orphan requirements are also identified and listed.



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

(4) Some of the proposed tests are not linked to functions listed in the WP4 deliverables. It is expected that subsequent updates will correct this.

We understand that any scenario that is not linked to any requirement may lack of a justification. However, most of the requirements are coming from all kinds of stakeholders, in particular experimenters and testbed owners; many of these requirements are high-level and most of them are addressed in validation scenarios. Verification tests may look sometimes disconnected from the actual requirements, but they represent important steps for the technical verification of a component or combination of components.

(5) The definition of success criteria is left as future work. This is reasonable for now but should be present in the next iteration.

Metrics and success criteria have been extensively reworked. Most of the success criteria are derived from the expectation and user/application requirements.

(6) This report must be revised to address the above deficiencies. The revision must identify clearly the way the deficiencies will be addressed in future iterations and those iterations must describe the solutions that were chosen.

The above deficiencies have been addressed and the corrective actions are briefly introduced in the above paragraphs.



## **Part III: Executive Summary**

This deliverable is based on the results of T4.1 for what concerns the definition of the testing and validation scenarios of the RAWFIE platform. It describes the test and validation methodology and it defines a set of test scenarios that will be performed in WP6 as well as the definition of the success criteria.

In D3.1, the end users have specified the RAWFIE requirements at all possible levels (component, system, etc.) and many categories (functional, non-functional, etc.). These requirements shall be met by the RAWFIE testbeds, with respect to their achievement or specific success criteria. It defines the minimum set of requirements to be met by the testbed and specifies the scenarios that are sufficient to validate the testbed, with respect to requirement subsets.

The test and validation scenarios deal with the global features of the RAWFIE system. They cover the test and validation of the Open interface framework, the interoperability of different sets of entities (testbeds, UxV, etc.) and the management of the RAWFIE federation.

The test scenarios are used during the system integration and testing, in particular of the different front-ends, middle-tier, non-functional services (e.g. storage) and the operational entities (e.g. UxV, testbeds, and environment). The validation covers the entire RAWFIE Federation life-cycle, but it focuses on the deployment and operation phases.

Verification takes place during the development (e.g. in the way of unit tests) and on completion of development (integration tests), before the system is delivered to the pilot users. The purpose of verification is to ensure that each component works as expected and RAWFIE prototype components are related correctly through all expected scenarios. The verification process also offers an opportunity to test RAWFIE under extreme conditions such as realistic volumes of data, to give an indication of theoretical performance and ensure that the system is scalable to a sufficient degree when it is deployed for the users.

In order to verify components, the Consortium has identified all components of the system and verification scenarios for each of them has been prepared. Verification needs to be carried out on each component by way of unit tests to be sure that the required functionality is achieved in the way that is expected, and on the whole system to ensure that it achieves the required functionality, performance and reliability.

Evaluation takes place once RAWFIE prototype has been deployed for the pilot users to assess how the system performs under live scenarios. Evaluation covers areas such as the usability of the user interfaces, the type, quantity and quality of the data provided and overall use and usability of the system. The system will be evaluated following the metrics defined in this document.



## **Part IV: Main Section**

### **2 Introduction**

#### **2.1 Scope of D4.6**

This deliverable specifies the verification and validation scenarios to be exercised on a RAWFIE testbed and the success criteria used for the evaluation of its implementation. Validation scenarios aim at checking if the system works as expected from the End Users point of view (System Validation). They can be refined and enhanced at a later stage in cooperation with WP6, and have to be strictly linked to the Use Cases defined within WP3. This document also prepares the approach for Components and Integrated Prototype Testing (System Verification) for Task 6.1 (e.g. functional and performance tests, and so on). Finally, it describes the Verification vs. Validation activities and approaches.

D4.6 should be used as input for the work on WP6, in particular:

- Task 6.1 Prototype Integration, Testing and customization
- Task 6.2 Evaluation and Platform Validation

The document covers:

- What needs to be tested (complete testbeds, subsystems, etc.);
- Who will test (users, stakeholders, RAWFIE partners, EAB, etc.);
- How tests are performed (tools, means, metrics, criteria, etc.).



## 2.2 Abbreviations

Table 1: Abbreviations

Abbreviation	Meaning
ACCS	Accounting Service
AT	Aerial Testbed
AUV	Autonomous Underwater Vehicle
BS	Booking Service
BT	Booking Tool
DoW	Description of Work
EAT	Experiment Authoring Tool
EC	Experiment Controller
ECV	EDL Compiler and Validator
EDL	Experiment Description Language
EMT	Experiment Monitoring Tool
EST	Early sub-system tests
LS	Launching Service
MT	Maritime Testbed
MM	Monitoring Manager
NC	Network Controller
PA	Platform Administrator
PT-DAA-E	Data Analysis Engine
PT-DAA-T	Data Analysis Tool
RC	Resource Controller
RET	Resource Explorer Tool
SYMS	System Monitoring Service
SMT	System Monitoring Tool
TD	Testbed Directory
TM	Testbed Manager
TO	Tesbed Operator
UAV	Unmanned Aerial Vehicle
UM	UxV Manufacturer
URS	Users & Rights Service
UD	User Defined
UGV	Unmanned Ground Vehicle
USV	Unmanned Surface Vehicle
UxP	UxV Proximity component
UxV	Unmanned aerial/ground/surface Vehicle
UxVNT	UxV Navigation Tool



VE	Visualisation Engine
VT	Vehicular Testbed
VT (scenario)	Visualisation Tool
WP	Web Portal
WT	Wiki Tool

### 3 Object of the validation and testing

The RAWFIE system is made of a set of sub-systems, components, processes, etc. and, thus, it should be thoroughly validated and tested. Only through an efficient verification and validation process, possible problems and malfunctions will be revealed and corrected in order to secure the efficient execution of the RAWFIE platform. A set of scenarios have been defined to verify the properties of the RAWFIE system during the development, to verify that the RAWFIE system and components comply with the specifications and to evaluate the degree of achievement with respect to the expected performance. The RAWFIE consortium aims to secure the efficient execution of the system in two axes: (a) the *verification* of the available components and the integrated system, (b) the *validation / evaluation* of the whole system.

The verification process aims at revealing potential problems. A set of template for describing components and system integration tests that must be passed (functional tests, performance tests, etc.) will be defined. Verification scenarios are adopted to verify that the platform and the single components (as implemented within WP5) properly meet the requirements from the technical perspective (system verification). The system validation and evaluation process aims to reveal if the system also meets the defined requirements and performs as expected from the end users perspective. Similarly to the verification process, the validation will be built on top of a set of templates for describing the validation scenarios. The establishment of the scenario descriptions and specifications was initially based on the analysis of the user requirements defined in D3.1 and the related metrics and expected performance (success criteria); the analysis of the proposals received by the consortium in the frame of the first RAWFIE Open Call revealed new use cases and scenarios, which were considered as additional “user defined” scenarios.

Nota bene: The template used for describing the scenarios already includes a “status” of the capability for RAWFIE to pass it, although the verification scenarios are defined for future experimentation. This field is currently a placeholder for the upcoming tests that will be performed, since we will complete these templates across the entire project lifetime and probably beyond it. Whenever the verification (or validation) will be done, we will update the status.

#### 3.1 Verification

Verification takes place during the development (e.g., in the way of unit tests) and on completion of development (integration tests) before the system is delivered to the pilot users.





The purpose of verification is to ensure that each component works as expected and RAWFIE prototype components are interacting correctly through all expected scenarios. The verification process also offers an opportunity to test RAWFIE under extreme conditions such as realistic volumes of data to give an indication of the theoretical performance and ensure that the system is scalable to a sufficient degree when deployed for the users. The aim is to answer questions related to if the developed components meet the initial requirements and if they are built in the right way. In order to verify the available components, the consortium has identified all components of the system and verification scenarios for each of them has been prepared. Verification needs to be carried out on each component by way of unit tests to be sure that the required functionality is achieved in the way that is expected, and on the whole system to ensure that it achieves the required functionality, performance and reliability. Verification will help to lower the number of defects in early as well as in late stages of development and lead to better understanding of the components. Finally, it will reduce the chances of failures in the software implementation.

### **3.2 Validation and evaluation**

Validation and evaluation takes place once the RAWFIE prototype has been deployed for the pilots to assess how the system performs under live scenarios. Evaluation covers areas such as the usability of the user interfaces, type, quantity and quality of the data provided and the overall use and the usability of the system. The system will be evaluated adopting the metrics defined in this document. The discussed process will execute extensive evaluations in order to assess the overall effectiveness and efficiency of the RAWFIE solution and to prove its added-value in a real environment. The validation campaign will include formal tests of the RAWFIE platform against the requirements set, as well as against the use cases' objectives. Validation sessions and templates, based on requirements will take place, expecting to bring valuable information about general user acceptance and usability of the provided infrastructure. Performance or other technical issues will be thoroughly evaluated. The activity will conclude with the preparation of a report summarizing the system evaluation and providing an assessment of its readiness for operational use.

### **3.3 RAWFIE federation lifecycle**

The RAWFIE federation lifecycle will be tested through specific scenarios that 'see' the framework as a black box. The aim is to identify if the system works appropriately through a high level evaluation. At first, the tests will identify if a set of different testbeds are smoothly attached to the RAWFIE architecture. The test scenarios will define the type, the number and the location of the testbeds. Accordingly, a specific EDL script will be defined that covers the entire set of the available components and testbeds. For instance, the script will define requirements for the parallel execution of different types of testbeds in the same experiment. In combination with the stress tests, the specific approach is judged very efficient as it will identify possible problems in the RAWFIE architecture. In general, the federation lifecycle will be evaluated through a number of major phases that include: user and testbed registration, authoring, booking, launching and evaluation of an experiment. In the upcoming



sections, a set of validation scenarios are provided that cover all the discussed phases accompanied by a set of metrics that will reveal the performance of the framework.

### 3.4 Verification and validation infrastructure and procedures

Verification will ensure that RAWFIE components meet the defined requirements while the validation phase will check if the system meets the high level requirements as defined by the consortium. Requirements are verified and the implemented components and the system are evaluated against the defined requirements. In addition, the validation process will ensure that all requirements are adequately tested or demonstrated, and that test results are as expected and can be repeated to verify correct implementation of the RAWFIE components. The consortium will follow a specific plan that follows these guidelines and it will help to ensure that the provided components can consistently meet a high level of quality and performance requirements. In short, the verification and the validation plans are as follows:

- **Verification plan.** For each component and sub-components the tests will manage to reveal their performance. Specific objectives will be defined for each (sub-) component and a detailed description of the verification scenario will be provided. Moreover, pre-requisites and the expected results will undertake the role of identifying if the component meets the defined requirements. Finally, specific testing scenarios could be devoted to identify the appropriate communication between components in order to secure the efficient data transfer throughout the RAWFIE architecture. The discussed plan will be realized during the implementation process in order to identify possible problems early in the development process.
- **Validation plan.** A set of validation scenarios will be adopted to reveal the performance of the platform. These scenarios mainly focus on testing from the stakeholder's point of view. Hence, in each scenario the main stakeholders will be defined and a detailed description will elaborate on the adopted steps. In addition, the involved (sub-) components will be referred in order to have a view on the part of the RAWFIE architecture that is evaluated. It should be noted that these scenarios will be evaluated against the already defined requirements.

#### 3.4.1 Non regression and stress tests

The aim of non-regression and stress tests is to identify possible errors in the RAWFIE architecture. These errors could be caused by a number of issues like wrong interfaces design and / or implementation, insufficient data passed to / from each component and so on.

Non regression tests will be realized on the RAWFIE prototype. As it is very difficult to have a large set of UxVs during validation, specific routines will undertake the role of producing data related to UxVs behaviour (e.g., location, measurements, status of resources). Hence, the consortium will be capable of performing large scale validation producing large amounts of data in high rates. The discussed routines will be launched / combined with the prototype and will represent the behaviour of RAWFIE nodes / testbeds. A post-processing tool will undertake the responsibility of analysing the derived behaviour of the system based on a set



of metrics. For instance, the number of errors, the data transferred, the time required to complete an ‘action’ and so on are some useful metrics that could be adopted to measure the performance of the system. In addition, the consortium will adopt an approach that will take into consideration the ‘footprint’ of each test. This means that every validation scenario will be combined with a specific ‘view’ of the system. For instance, specific tests will be realized either from the experimenter point of view or from the testbed perspective. In other words, the ‘footprint’ will combine each test with what is tested (i.e., RAWFIE architecture). Finally, specific reports will be realized to describe the outcome of the process.

Based on the aforementioned routines, the consortium will have the opportunity to provide extensive tests in order to reveal the performance of the platform. The aim is to bring the framework close to its limits. Fails and means for fast recovering will be realized leading to a high quality system. Stress tests will be realized in the following axes: (a) high number of users (b) high number of bookings, (c) high number of concurrent connections to the system, (d) high number of testbeds / nodes, (e) high load, (f) unpredictable events like taking a testbed / node or the DBMS offline and restarting it, etc. These tests focus on unpredictable events randomly generated during the framework execution and put emphasis on robustness, availability, and error handling under a heavy load, rather than on what would be considered correct behaviour under normal circumstances.

### 3.4.2 EDL Testing

The EDL testing is a special process in the verification – validation process. The reason is that EDL tests should reveal the efficiency of the system when communicating with experimenters not only through the provided functionality perspective but also through the easiness that an experimenter can create, compile and run an experiment. The aim of the EDL testing is to reveal if the scenario defined by the experimenter is smoothly processed and produces the appropriate outcomes to be adopted by the remaining RAWFIE components. Specific tests will be realized concerning important characteristics of the EDL as well as the functionalities provided by the editors. For instance, the testing process will involve two aspects: (a) the experimenter side and (b) the components side. From the experimenter point of view, the provided editors and their functionalities should be easily initiated and commands (i.e., EDL scripts) should be efficiently translated based on the underlying EDL model. In RAWFIE, experimenters that create an experiment will need to provide a short high level description of the experiment and its purpose. The second aspect involves the definition of specific commands in the test script that will reveal if the RAWFIE components are smoothly combined. This will also test the connection between components in order to have an efficient execution of the experiment.

The test scenarios will be realized based on the defined use cases and reveal if an experimenter is capable of easily define an experiment in the EDL terms. For instance, with test scenarios, critical questions will be answered like: Can the experimenter easily define the application logic of his/her experiment? How easily the experimenter can define an experiment that realizes a complex algorithm? Moreover, the test scenarios will check if the EDL script is efficiently translated based on the underlying model and, accordingly, be



compiled and validated. Syntactic and semantic errors will be incorporated in the test scenarios in order to reveal if the system is capable of identifying the errors and return specific messages to the experimenter. Successful fulfilment of the compilation and the script validation process will be realized through a number of files / models assigned to specific RAWFIE components. These files / models are necessary to, finally, execute the experiment.

## 4 Stakeholders and actors

Stakeholders to be considered in the validation plan, include both end users that are interested in the experimentation of specific technologies, as well as personnel of specialised, NGO or GO organisations, that can use the RAWFIE platform and testbeds for simulating specific mission scenarios linked to their by day operations. All these types of actors, some of them identified in D3.1/D3.2, are in the following represented by the common category “Experimenters”. Those who are the main candidate for evaluating the appropriateness of the RAWFIE platform and testbeds to support their requirements are:

- Experimenters:
  - Users who belong to the federation. They must be acknowledged by the federation partners. As said, these include different stakeholders like e.g.:
    - Governmental Organizations responsible for SAR operations
    - Non-Governmental Organizations aiding SAR operations
    - Command and Control Operation centres
- RAWFIE Admin:
  - Administrator of RAWFIE frontend and middleware framework. These are owned and maintained by the RAWFIE consortium
- Testbed Operators:
  - Owners and managers of testbed facilities
- UxV Manufacturers:
  - Suppliers of UxVs resources

## 5 Metrics (IES)

### 5.1 Introduction

In the current version of D4.6, the list of validation metrics has been updated in the following directions:

- considering the type of actors mentioned in the previous section



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

- modifications of some metrics in order to include the link to the related D3.2 requirements (and in very few cases to some provided in D3.1), where applicable
- alignment of metric types to the requirements' types as defined in D3.2: (PERF=performance, FUNC=functional, USE=usability, DATA=data). It has to be noted however, that metrics linked to one or more D3.2 requirements are not necessarily of the same type of the requirement/s they are linked to. A specific validation metric could be, for example, of type USE because focussed on usability from the validators (end users) perspective, while being connected to functional requirements
- modifications of the metrics in order to be more specific and / or to better reflect the related functionalities and design as expressed in the new requirements and components' design deliverables D3.2 and D4.5
- addition of the success criteria as already planned in the previous deliverable
- removal of some of the old metrics, by taking into account that the validation process is about understanding whether “we are building the right system”, mainly from the end users' perspective. Therefore by focussing, mainly, the validation metrics on end users' needs and perception (perceived platform usability, performances, stability, accessibility, and understandability)

As anticipated, some of the validation metrics are derived from the list of Requirements described in D3.2, as they are strictly related to system and users' requirements.

The following metrics categories are taken into account:

- PLATFORM – metrics related to the whole RAWFIE frontend and middleware platform behaviour
- TESTBED – metrics related to testbeds availability / information
- INTERCONNECTIVITY – metrics related mainly to communication performances
- UxV - metrics related to UxVs availability / information

Once the system has been verified and deployed at the testbed sites, a period of evaluation will take place during which the abovementioned metrics will be assessed either by quantifiable measurements or by way of questionnaires/interviews. It should be noted that not all of the defined validation metrics can be directly and explicitly expressed in the validation scenarios described in the following of the document.

Nevertheless, the needed actions will be put in place for being able to measure them and evaluate them against the related success criteria. This applies for example, to the metrics related to the monitoring and acquisition of particular parameters / statistics (errors, notifications, etc.), as well as to most of the usability related ones (type = USE), for which dedicated questionnaires will be prepared before running the validation sessions, to be submitted to the validators.



For the description of other specific metrics' attributes like required or beneficial, hard or soft, please refer to the previous deliverable D4.3.

## 5.2 Metric template definition

Below is the new, updated version of the metrics definition / description template

**Table 2: Metrics template**

Metric category/ Type/ ID / Tag	Description	Required or Beneficial Hard or Soft	Mean for measurement	Validator stakeholder	Success Criteria	Req. Id (D3.1- D3.2)

## 5.3 Success criteria

Success criteria are quantitative or qualitative values (or set or ranges of values) for relevant metrics, against which the actual characteristics or performance indicators of the system and components are compared. A typical criterion is a threshold against which the performance indicator of the tested element is compared (e.g. “the temperature of the motor shall not exceed 90°C during the experiment”).

The success criteria are usually combined to perform the evaluation of a given element. For example, an element will be successfully evaluated if it meets the criteria A and B and C. Another element may be successfully evaluated if it meets the criteria B and C or F.

For any given metrics, the success criteria may vary depending on the components under evaluation, or on the experiment under execution. To this intent, a template is provided to specify criteria for any component or system to be evaluated.

## 5.4 Platform metrics



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

Metric category/ Type / ID / Tag	Description	Required or Beneficial Hard or Soft	Mean for measurement	Validator stakeholder	Success Criteria	Req. Id (D3.1-D3.2)
PLATFORM / PERF	Measures the performances of the system as a whole according to specific sub-criteria described in the following					
PLATFORM / PERF / 1 / STABLE SYSTEM	Measures the system uptime and detect system downtimes	Required Hard	System monitoring System logs	RAWFIE Admin, Testbed Operator, Experimenter	Downtime < 2%	PT-SYM-T-001 PT-SYM-T-004
PLATFORM / PERF / 2 / ERRORS	Counts RAWFIE platform errors and crashes	Required Hard/Soft	System monitoring System logs Tickets received from the end users	RAWFIE Admin Experimenter	The target is to keep the number of received tickets to the minimum possible, i.e. under a threshold of the 5% of the total number of executed experiments	PT-EXP-C-009
PLATFORM / PERF / 3 / SCALABILITY	Number of concurrent running experiments. Number of users interacting with the platform (e.g. for creating experiments, visualise and analyse results, and so on.	Required Hard	System statistics and monitoring (e.g. users' accesses). Stress tests by launching a certain number of experiments (the maximum allowed by the available testbeds & resources) in parallel. Registering the number of successfully executed experiments, and date/time of execution.	RAWFIE Admin	By design, the target (success criteria) is to have a platform that can scale horizontally, provided the needed server instances are setup in the Cloud environment. Therefore this metric should only be dependent on the number of available testbeds and UxVs at each given time	PT-NF-006 from D3.1



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

<p>PLATFORM / PERF / 4 / RECOVERY TIME</p>	<p>Records the time needed to recover the system operations after the shutdown / failure of specific parts which are needed for normal system use (i.e. Web frontend server, Middle Tier server/services, Message Bus servers cluster, and so on).  Testbeds and UxVs unavailability is excluded as they are independent from the central platform</p>	<p>Required Hard</p>	<p>System monitoring Statistics, collected through dedicated tests for simulating the unavailability of specific services / servers</p>	<p>RAWFIE Admin</p>	<p>The system should be operational again after one or more server / services shutdown, in less than 5 minutes.  This should happen either automatically (thanks to the used cloud facilities and setup), or event manually in case of problems affecting the functionality of specific services, requiring a technical intervention.  In this latter case, the time is calculated starting from when the problem causing the shutdown of the server / service has been solved, and the operator himself has started again the affected servers / services</p>	<p>PT-SYM-S-004</p>
<p>PLATFORM / PERF / 5 / LATENCY/ RESULTS UPDATE TIME</p>	<p>Latency between the real execution of commands or the acquisition of measurements and results, and the update of the same info in the visualisation tools</p>	<p>Required Hard</p>	<p>System monitoring and statistics / logs</p>	<p>RAWFIE Admin Experimenter</p>	<p>&lt; 5 seconds</p>	





## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

PLATFORM / PERF / 6 / LATENCY/ BOOKING TIME	Time for the user to receive the notification of “experiment booked” after completing the request procedure through the UI (e.g. completion of all queries for selecting the needed testbeds and resources also in a federated environment).	Required / Hard	System monitoring and statistics / logs	RAWFIE Admin Experimenter	< 30 seconds	
PLATFORM / USE	Measures the usability of the system as a whole, or of different GUI tools and functions, according to specific sub-criteria (provided notifications, ease of access, clarity, engagement, motivation, etc..) described in the following					PT-WEB-P-001
PLATFORM / USE / 7 / NOTIFICATION	Measures the quality and usefulness of the notifications provided by the different system tools	Required Soft	End users’ questionnaires / interviews, aimed at checking whether the notifications provided by specific GUI tools, are understandable and properly provided.	Experimenter	Through the answers to specific questions (for each GUI tool), users will be asked to give a score from 1 to 5. The metric will be considered as positively evaluated if an average score of at least 3.5 will be reached.	PT-BOO-T-010 PT-BOO-T-010 PT-EXV-S-001 PT-BOO-S-011 PT-LAU-S-008 PT-LAU-S-012 PT-EXP-C-008 PT-EXP-C-009



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

PLATFORM / USE / 8 / ROLES	RAWFIE platform shall support various roles with different privileges at every level of access	Required Soft	End users' questionnaires / interviews, aimed at checking whether the role management is provided as end users' expect.	Experimenter	Through the answers to specific questions (for each GUI tool), users will be asked to give a score from 1 to 5. The metric will be considered as positively evaluated if an average score of at least 3.5 will be reached.	PT-WEB-P-002 PT-SYM-T-003 PT-USR-S-001 PT-USR-S-002
PLATFORM / USE / 9 / VISUALISATION / BALANCE	End user estimate the distribution of the optical weight in the GUI (number of objects) in a picture via questionnaires	Required Soft	Questionnaire	Experimenter	Through the answers to specific questions (for each GUI tool), users will be asked to give a score from 1 to 5. The metric will be considered as positively evaluated if an average score of at least 3.5 will be reached.	PT-BOO-T-009
PLATFORM / USE / 10 / VISUALISATION / SIMPLICITY	Experimenter evaluates if the objects appearing to the screen are the minimum needed and easily accessible	Required Soft	Questionnaire	Experimenter	Through the answers to specific questions (for each GUI tool), users will be asked to give a score from 1 to 5. The metric will be considered as positively evaluated if an average score of at least 3.5 will be reached.	PT-BOO-T-009 PT-VIS-E-002
PLATFORM / USE / 11 / VISUALISATION / CONSISTENCY	Experimenter evaluates if similar actions lead to similar results and the elements in the GUI (fonts, patterns, tables) are similar to all pages	Required Soft	Questionnaire	Experimenter	Through the answers to specific questions (for each GUI tool), users will be asked to give a score from 1 to 5. The metric will be considered as positively evaluated if an average score of at least 3.5 will be reached.	PT-BOO-T-009 PT-LAU-S-003



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

PLATFORM / USE / 12 / VISUALISATION / UTILITY	Experimenter evaluates the utility of the different tools in order to define, manage and execute an experiment	Required Soft	Questionnaire	Experimenter	Through the answers to specific questions (for each GUI tool), users will be asked to give a score from 1 to 5. The metric will be considered as positively evaluated if an average score of at least 3.5 will be reached.	PT-EXA-T-002 PT-VIS-E-002
PLATFORM / USE / 13 / GUIDANCE	Experimenter tests if help guidance or error messages appear in order to guide him/her to the right option	Required Soft	Questionnaire	Experimenter	Through the answers to specific questions (for each GUI tool), users will be asked to give a score from 1 to 5. The metric will be considered as positively evaluated if an average score of at least 3.5 will be reached.	PT-EXA-T-002
PLATFORM / USE / 14 / FILTERING	Usefulness and efficiency of provided filtering functionalities of the different tools	Required Soft	Questionnaire	Experimenter	Through the answers to specific questions (for each GUI tool), users will be asked to give a score from 1 to 5. The metric will be considered as positively evaluated if an average score of at least 3.5 will be reached.	PT-SYM-T-003 PT-REE-T-003 PT-EXA-T-006 PT-VIS-T-005 PT-DAA-T-004 PT-DIR-S-002
PLATFORM / USE / 15 / EXPERIMENTS STATISTICS	It should be possible to check if the same or similar experiment configuration (parameters) lead to problems (UxV collisions, crashes, system failures, etc.) in the past	Beneficial Hard/Soft	System monitoring, logs, questionnaire	RAWFIE Admin, Experimenter	RAWFIE Admin validate the quality and quantity of provided information from past experiment.  Through the answers to specific questions (for each GUI tool), users will be asked to give a score from 1 to 5. The metric will be considered as positively evaluated if an average score of at least 3.5 will be reached.	PT-DAA-T-002



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

PLATFORM/ FUNC/16/ STORAGE	System ability to store experiment data in case of comm. link failure between the testbed and the upstream components deployed in the cloud	Required Hard	System Monitoring, logs Check stored data	RAWFIE Admin	The system should be able to provide, for visualisation and analysis purposes, all (100%) results related to the experiments that were running when the link communication failure happened	PT-GEN-R-004 PT-VIS-E-004
PLATFORM/ FUNC/17/ EXTENSIBILITY	This metric is aimed at assessing how easy is to extend the platform in terms of: A) new services / functionalities; B) New testbeds and UxVs provided the architectural guidelines and requirements are respected by new testbed and UxVs owners, and with or without (SFA based) federation	Required Soft/Hard	Conceptual evaluation by RAWFIE technicians and stakeholders	RAWFIE Admin Testbed Operators UxVs Manufactures	The different architectural elements (from Frontend Tier to MiddleTier services to testbeds and UxVs) should be easily “plugged”, from the software perspective, with the minimum effort, by just using configuration capabilities and APIs that are provided by the RAWFIE platform components themselves	

**Table 3: Platform metrics.**



## 5.5 Testbed metrics

Table 4: Testbed metrics

Metric category/ Type / ID / Tag	Description	Required or Beneficial  Hard or Soft	Mean for measurement	Validator stakeholder	Success Criteria	Req. Id (D3.2)
TESTBED / DATA / 1 / INFORMATION	Capability of a testbed to provide the users, through the RAWFIE platform, information relevant for booking and running experiments, such as: weather conditions, UxV availability and capabilities, sensors, whole testbed availability time	Required Hard/Soft	Testbed monitoring (and finally notifications to the users)  Users' questionnaires	RAWFIE Admin, Experimenter	Weather conditions, overall testbed status, as well as information on UxVs and sensors, should be updated at least <b>daily</b> by the Testbed Operator <u>during the periods when the Testbed is up and running.</u>  And made available for the experimenter the 100% of the time.	TB-MOM-001 TB-MOM-002 TB-MOM-003 TB-MOM-004 PT-EXP-C-006 PT-EXP-C-008 PT-SYM-002 TB-GEN-R-001 TB-GEN-002 TB-MAN-003
TESTBED / FUNC / 2 / SECURITY	Capability of the Testbed to provide a secure environment, with firewall rules for avoiding harmful accesses to the rest of the RAWFIE platform. It could also be based on a DMZ containing only the Testbed components which need to be reached from the rest of RAWFIE components	Required Hard	Dedicated security tests	Admin	The success criteria is defined as the set of rules that will need to be satisfied in order to avoid unauthorised accesses to the different components, at both RAWFIE platform and Testbed side.	TB-PRO-002



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

TESTBED / FUNC / 3 / AVAILABILITY	Measure the Testbeds availability for performing experiments, in a certain period of time	Required Hard	System monitoring & notifications. Users' experience	Admin, Experimenter	Success criteria will be that the amount of days of testbed availability in total, will be exactly as declared by RAWFIE Testbed Operators at the beginning and in the Open Calls proposals. Downtime for maintenance, as well as other planned unavailability which may prevent the execution of the experiments should be communicated in advance, at least 2 days before.	
TESTBED / USE / 4 / CONSISTENCY	This metric is intended to measure if the remote users of the scenario were able to perform their tests as they expected (e.g. the run experiment was exactly what they asked for)	Required Soft	Users' experience	Experimenter	Through the answers to specific related questions, users will be asked to give a score from 1 to 5. The metric will be considered as positively evaluated if an average score of at least 3.5 will be reached.	PT-EXP-C-009



### 5.6 UxV metrics

Metric category/ Type / ID / Tag	Description	Required Or Beneficial Hard Or Soft	Mean for measurement	Validator stakeholder	Success Criteria	Req. Id (D3.2)
UxV / FUNC / 1 / COHERENCE	Actual route vs. plan	Required Hard/Soft	Statistics of the UxV collected during the experiment	RAWFIE Admin, Experimenter	Through the answers to specific related questions, users will be asked to give a score from 1 to 5 to this metric. The metric will be considered as positively evaluated if an average score of at least 3.5 will be reached.	TB-REC-003 TB-REC-004 TB-REC-005
UxV / FUNC/ 2 / MISSION ACHIEVEMENT	Actual mission achievement	Required Hard/Soft	Experiment statistics: rate of achieved vs. assigned objectives	RAWFIE Admin, Experimenter	Through the answers to specific related questions, users will be asked to give a score from 1 to 5 to this metric. The metric will be considered as positively evaluated if an average score of at least 3.5 will be reached.	TB-REC-003 TB-REC-004 TB-REC-005
UxV / PERF / 4 / BATTERY LIFETIME	Counts battery lifetime per experiment	Required Hard	System Monitoring. UxV node parameters and status	RAWFIE Admin, UxV Manufactors	Battery autonomy of each device should be between 15 and 30 minutes	UXV-NOD- 002

Table 5: UxV metrics

### 5.7 Interconnectivity (aka. communication) metrics

Communication metrics are related to traditional networking and communication parameters like throughput, end-to-end delay (latency), and maximum allowed communication distance with the described below might be applied both to the *local* communication between the UxVs and the Resource Controller at the testbed side, as well as the remote communication with the rest of RAWFIE platform.



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

Metric type/ ID/ Tag	Description	Required or Beneficial Hard or Soft	Mean for measurement	Validator stakeholder	Success criteria	Req. Id (D3.2)
INTERCONNECTIVITY / PERF / 1 / AGGREGATED THROUGHPUT	Aggregated data throughput for the whole RAWFIE platform, expressed as the maximum amount of messages processed in the unit of time	Required Hard	System monitoring. Components measurements. By the mean of stress tests, messages of different fixed size (e.g. typical average sized RAWFIE messages) will be processed for a given workflow (e.g. a given validation scenario). At the end of the test, the total processed number of messages in the given amount of time is retrieved, and the conversion in bytes per second is finally realised	RAWFIE Admin Testbed Operator (for performing validation scenarios)	The actual, acceptable throughput for the correct execution of realistic experimentation scenarios is part of the research activities. The validation will, in this case, aimed at A) assessing the performances of the provided integration and communication solution  The aggregated throughput will be calculated for different workflows (e.g. corresponding to some of the validation scenarios)	
INTERCONNECTIVITY / PERF / 1 / COMPONENTS THROUGHPUT	Data throughput ensured by different RAWFIE components, for both the intra-testbed communication (especially Resource Controller-to-UxVs) and inter-tier communication	Required Hard	System monitoring. Components measurements. By the mean of stress tests, messages of different fixed size (e.g. typical average sized RAWFIE messages) will be processed for a given communication scenario (e.g. between 2 components in a validation scenario). At the end of the test, the total processed number of messages in the given amount of time is retrieved, and the conversion in bytes per second is finally realised	RAWFIE Admin Testbed Operator	See the previous metric	





## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

<p>INTERCONNECTIVITY / PERF / 2 / END-TO-END DELAY</p>	<p>The total time it takes for a packet to reach its destination after being sent. Especially relevant for the communication between the resource controller and the UxVs (local, testbed level), but in general for any other kind of components' communication scenario</p>	<p>Required Hard</p>	<p>Stress tests are performed by continuously sending packets of fixed size (e.g. average size of RAWFIE messages exchanged between the Resource Controller and the UxVs). Each packet is sent with a timestamp (the sender and the receiving entities (where the involved components are running are synchronised with the same time). At the receiver side, for each received packet, the difference between the receiving time and the original timestamp is calculated</p>	<p>RAWFIE Admin Testbed Operator</p>	<p>The actual, acceptable end-to-end delay for UxV controlling is part of the research activities. The validation will, in this case, aimed at A) assessing the performances of the provided integration and communication solution and B) finding outcomes of the impact of the latency in semi-autonomous devices controlling</p>
--	---	--------------------------	--	--	---

Table 6: Communication metrics

## 6 Verification

The verification of components is included in this chapter in an attempt to capture, from the earliest stage of the project, as most input as possible discussing the scenarios and tests about the verification and validation.



## 6.1 Verification scenarios

### 6.1.1 Frontend Tier (Web Portal GUI elements)

Table 7: Verification test of the Web Portal - Login/ Logout

Test ID: <b>WP01</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Web Portal - Login/ Logout</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>User registered in the User &amp; Rights repository</li> </ul>		
<b>Related Requirements</b>		PT-WEB-P-001, PT-WEB-P-002		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	user opens RAWFIE any web page	redirect to login page, login form displayed		
2	user enters invalid credentials and submits the form	error message displayed		
3	user enters valid credentials and submits the form	redirect to start page		
4	user press the logout button	redirect to login page, login form displayed, logout message displayed		

Table 8: Verification test of the Web Portal – Language selection

Test ID: <b>WP02</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Web Portal – Language selection</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Translation available</li> </ul>		
<b>Related Requirements</b>		PT-WEB-P-001		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	user opens RAWFIE any web page	web page with language selection displayed,		
2	user changes the language	web page displayed in the selected language		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

Test ID: <b>WP03</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Web Portal – User management</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>• Admin login available</li> <li>• No pending registration request</li> </ul>		
<b>Related Requirements</b>		PT-WEB-P-002		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Browser 1: login as administrator and open user management page	management page displayed		
2	Browser 1: Navigate to registration requests page	No registration request displayed		
3	Browser 2: Open register form, fill in form (login credentials, personal data, etc.) and submit	Registration request stored and confirmation shown to the user.		
4	Browser 2: Try to login with the submitted login credentials	Login failed. Display message that user is locked		
5	Browser 1: Reload registration requests page	The new registration request is show		
6	Browser 1: Accept the new user	The new user is now unlooked		
7	Browser 2: Try to login with the submitted login credentials	Login successful.		
8	Browser 1: Navigate to the user list and delete the new user	User deleted		
9	Browser 2: Logout and try to login with the submitted login credentials	Login failed. Show invalid credentials messages		



6.1.1.1 Wiki Tool

Table 9: Verification test of the Wiki Tool – Component Help

Test ID: <b>WT01</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Wiki Tool – Component help</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>• Help pages added to the Wiki</li> </ul>		
<b>Related Requirements</b>		PT-WEB-P-003		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Login to the Web Portal and open Resource Explorer	Resource Explorer page displayed		
2	Click on the Help icon	Wiki Tool opened with the article about Resource Explorer		
3	Repeat step 2 of other pages (like Visualization Tool, Booking tool, etc.)			

Test ID: <b>WT02</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Wiki Tool – Editing</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>• User for Wiki management defined</li> </ul>		
<b>Related Requirements</b>		PT-WEB-P-003		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Login to the Web Portal as normal experimenter and open a page in the Wiki Tool	Wiki page displayed		
2	Try to edit the page	Editing not possible due to missing rights		
3	Login as administrator and assign the Wiki manager right to the user	The user has now the Wiki manager right		
4	Login as the first user and open a page in the Wiki Tool	Wiki page displayed		
5	Try to edit the page	Editing allowed as changes are save		



6.1.1.2 Resource Explorer Tool

Table 10: Verification test of the Browse testbeds and UxVs and start booking

Test ID: <b>RET01</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Browse testbeds and UxVs and start booking</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>• connection to the Testbeds Directory Service OK</li> <li>• data about testbeds and UxVs available</li> </ul>		
<b>Related Requirements</b>		PT-REE-T-001, PT-REE-T-003, PT-REE-T-004		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	user opens Resource Explorer Tool in the Web Portal	Resource Explorer Tool displays a view with all available testbeds		
2	user selects a testbed	Resource Explorer Tool displays all testbed details and a list of available UxVs		
3	user selects a UxV	Resource Explorer Tool displays all UxVs details		
4	user starts booking	Booking Tool opened with the selected resources		

6.1.1.3 Booking Tool

Below tests related to verifying Booking Tool correct behaviour and adherence to requirements defined in D3.2 are provided. The exact requirements addressed by the tests are provided in the Related Requirements field of the testing card.

Booking Tool requirements not addressed by the tests specified below

- PT-BOO-T-002,
- PT-BOO-T-011,
- PT-BOO-T-012



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 11: Verification test of the Booking Tool Calendar View and its display options**

Test ID: <b>BT01</b>	Conducted by:	Date:	Test Category: <b>Verification Tests (web tier)</b>	
<b>Hardware Configuration</b>	-			
<b>Software Configuration</b>	-			
<b>Test Name:</b>	<i>Booking Tool Calendar View and display options</i>			
<b>Preconditions</b>	<ul style="list-style-type: none"> <li>• connection to the Booking Service ok</li> <li>• user has logged in the web portal</li> <li>• reservations of different status exist in the Master DB</li> </ul>			
<b>Related Requirements</b>	PT-BOO-T-001 PT-BOO-T-003 PT-BOO-T-006 PT-BOO-T-010 PT-BOO-S-008			
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Click of Bookings menu item	Navigation to Booking Tool (Calendar View)		
		Calendar view displays by default the present week with all defined bookings		
2	Switch Calendar display to display week, month, day interval via the appropriate options	Calendar view changes to present the selected interval with all defined bookings		
3	Navigate back and forth in time via the provided navigation buttons (for every selection made in step 2)	Calendar view changes to previous or future date time intervals		
4	Verify by inspection of existing reservations that only reservations of certain status are visible in the Calendar View	Reservation of status PENDING, OK or REJECTED should only be displayed		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 12: Verification test of the Booking Tool Calendar View Interactions**

Test ID: <b>BT02</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (web tier)</b>
<b>Hardware Configuration</b>		-		
<b>Software Configuration</b>		-		
<b>Test Name:</b>		<i>Booking Tool Calendar View Interactions</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>• connection to the Booking Service ok</li> <li>• user has logged in the web portal</li> <li>• reservations of different status exist in the Master DB</li> </ul>		
<b>Related Requirements</b>		PT-BOO-T-001 PT-BOO-T-003 PT-BOO-T-005 PT-BOO-T-006 PT-BOO-S-002 PT-BOO-S-004		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Click on an empty calendar timeslot (result should depend on the relevance of the timeslot to the present time)	If click occurs on a past timeslot a popup warning is displayed		
		If click occurs on a future timeslot the "Create Reservation" window opens		
2	Click on an existing reservation (result should depend on the relevance of the reservation to the present time)	If click occurs on a past reservation the "Edit Reservation" window opens but no further actions are offered to the user		
	(see also test BT04)	If click occurs on a future reservation the "Edit Reservation" window opens and the user can perform certain actions on the reservation. Displayed actions depend on user role and reservation status		
3	verify the displayed color for each reservation (click existing reservations)	Coloring of reservation should differ based on the reservation status (shown in the Edit Reservation window)		



**Table 13: Verification test of the Booking Tool Create Reservation**

Test ID: <b>BT03</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (web tier)</b>
<b>Hardware Configuration</b>		-		
<b>Software Configuration</b>		-		
<b>Test Name:</b>		<b><i>Booking Tool Create Reservation</i></b>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>• connection to the Booking Service ok</li> <li>• user has logged in the web portal</li> <li>• user has clicked on an empty future timeslot</li> </ul>		
<b>Related Requirements</b>		PT-BOO-T-001 PT-BOO-T-003 PT-BOO-T-004 PT-BOO-T-009 PT-BOO-T-010 PT-BOO-S-006		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	User edits the field of the “Create Reservation” form so that no time overlapping with other reservation exists and presses the OK button (no conflicts scenario)	Reservation is created and displayed in the Calendar View. Reservation is put in PENDING state		
2	User edits the field of the “Create Reservation” form so that a time overlapping with other reservation exists and presses the OK button (possible conflict scenario)	If no common resources exist with the overlapping reservation then the new reservation is created and displayed in the Calendar View. Reservation is put in PENDING state		
		If common resources exist with the overlapping reservation then the new reservation is not created and a warning message is displayed		Result may depend on status of pre-existing reservation





## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 14: Verification test of the Booking Tool Edit Reservation Actions**

Test ID: <b>BT04</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (web tier)</b>	
<b>Hardware Configuration</b>		-			
<b>Software Configuration</b>		-			
<b>Test Name:</b>		<b><i>Booking Tool Edit Reservation Actions</i></b>			
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>• connection to the Booking Service ok</li> <li>• user has logged in the web portal</li> <li>• user has clicked on an existing future reservation</li> </ul>			
<b>Related Requirements</b>		PT-BOO-T-003 PT-BOO-T-005 PT-BOO-T-007 PT-BOO-T-008 PT-BOO-T-010 PT-BOO-S-006 PT-NF-002			
<b>Tools Used</b>					
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>	
1	The actions available to the Edit Reservation window depend on the: <ul style="list-style-type: none"> <li>• status of reservation</li> <li>• user</li> <li>• role of the user</li> </ul>				
	status=PENDING user= owner of reservation role= EXPERIMENTER	Actions available: OK, CANCEL DELETE			
	status=OK user= owner of reservation role= EXPERIMENTER	Actions available: OK, CANCEL DELETE			
	status=REJECTED user= owner of reservation role= EXPERIMENTER	Actions available: OK, CANCEL DELETE			
	status=PENDING user= owner of reservation role= TESTBED_OP	Actions available: OK, CANCEL, DELETE, APPROVE, REJECT			
	status=PENDING user= not owner of reservation role= TESTBED_OP	Actions available: CANCEL, APPROVE, REJECT			
	status=OK user= owner of reservation role= TESTBED_OP	Actions available: CANCEL, DELETE, REJECT			
	status=OK user= not owner of reservation role= TESTBED_OP	Actions available: CANCEL, REJECT			
	status=REJECTED user= owner of reservation role= TESTBED_OP	Actions available: CANCEL, DELETE, APPROVE			
	status= REJECTED user= not owner of reservation role= TESTBED_OP	Actions available: CANCEL, APPROVE			
	user= not owner of reservation	No actions available			
2	Owner of reservation performs changes to the reservation and	If the changes do NOT introduce conflicts in both timeslots and			



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

	presses OK button	selected resources then the reservation is successfully updated and the UI refreshed to display the changes		
		If the changes do introduce conflicts in both timeslots and selected resources then a warning message appears and no further action is performed		
3	Owner of reservation presses DELETE button	If reservation does not refer to a currently running experiment then it is put in a CANCELLED state and removed from the UI		
4	User with TESTBED_OP role presses APPROVE button	If no resource conflicts with already created reservation exists then reservation status becomes OK and color changes appropriately in the Calendar view		
5	User with TESTBED_OP role presses REJECT button	reservation status becomes REJECTED and color changes appropriately in the Calendar view		



6.1.1.4 Experiment Authoring Tool

Table 15: Verification test of the in-Textual Editor Experiments definition

Test ID: <b>EAT01</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier – middle tier)</b>
<b>Hardware Configuration</b>		-		
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Define Experiments in the Textual Editor</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>User entered in the RAWFIE Portal</li> </ul>		
<b>Related Requirements</b>		PT-EXA-T-001, PT-EXA-T-002, PT-EXA-T-003, PT-EXA-T-004, PT-EXA-T-005, PT-EXA-T-008, PT-EXA-T-009, PT-EXA-T-010, PT-EXA-T-011, PT-EXA-T-012, PT-EXA-T-013, PT-EXA-T-015		
<b>Tools Used</b>				
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1	Access to the Textual Editor through the RAWFIE Web Portal	Redirection to the Textual Editor interface		
2	Write an experiment	Experiment is presented in the editor		
3	Utilize code completion, content assist and compilation	The editor responds with specific drop down lists, messages, etc.		
4	Define erroneous commands in the experiment workflow	The editor responds with error messages and indication for correcting the error		
5	Save the experiment	The experiment is stored in the database and specific files are produced to be adopted by the remaining RAWFIE components		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 16: Verification test of the Textual Editor Experiments Update**

Test ID: <b>EAT02</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier – middle tier)</b>
<b>Hardware Configuration</b>		-		
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Update Experiments in the Textual Editor</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>User entered in the RAWFIE Portal</li> </ul>		
<b>Related Requirements</b>		PT-EXA-T-001, PT-EXA-T-002, PT-EXA-T-003, PT-EXA-T-004, PT-EXA-T-005, PT-EXA-T-008, PT-EXA-T-009, PT-EXA-T-010, PT-EXA-T-011, PT-EXA-T-012, PT-EXA-T-013, PT-EXA-T-015		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Access to the Textual Editor through the RAWFIE Web Portal	Redirection to the Textual Editor interface		
2	Open an already defined experiment	Experiment is presented in the editor		
3	Makes changes in the experiment workflow	The experiment is updated		
4	Save the experiment	The experiment is stored in the database and specific files are produced to be adopted by the remaining RAWFIE components		



Table 17: Verification test of the in-Visual Editor Experiments Define

Test ID: <b>EAT03</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier – middle tier)</b>
<b>Hardware Configuration</b>		-		
<b>Software Configuration</b>		•		
<b>Test Name:</b>		<i>Define Experiments in the Visual Editor</i>		
<b>Preconditions</b>		• User entered in the RAWFIE Portal		
<b>Related Requirements</b>		PT-EXA-T-001, PT-EXA-T-002, PT-EXA-T-003, PT-EXA-T-004, PT-EXA-T-005, PT-EXA-T-008, PT-EXA-T-009, PT-EXA-T-010, PT-EXA-T-011, PT-EXA-T-012, PT-EXA-T-013, PT-EXA-T-015		
<b>Tools Used</b>		•		
Step	Action	Expected Result	Status	Remarks
1	Access to the Visual Editor through the RAWFIE Web Portal	Redirection to the Visual Editor interface		
2	Access the available toolbar	Specific windows are presented		
3	Create an experiment by utilizing the available tools	The experimenter can defined waypoints and experiment information by clicking and designing in the visual editor		
4	Define erroneous commands	The authoring tool responds with error messages and indication for correcting the error		
5	Save the experiment	The experiment is stored in the database and specific files are produced to be adopted by the remaining RAWFIE components		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 18: Verification test of the in-Visual Editor Experiments Update**

Test ID: <b>EAT04</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier – middle tier)</b>
<b>Hardware Configuration</b>		-		
<b>Software Configuration</b>		-		
<b>Test Name:</b>		<i>Update Experiments in the Visual Editor</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>User entered in the RAWFIE Portal</li> </ul>		
<b>Related Requirements</b>		PT-EXA-T-001, PT-EXA-T-002, PT-EXA-T-003, PT-EXA-T-004, PT-EXA-T-005, PT-EXA-T-008, PT-EXA-T-009, PT-EXA-T-010, PT-EXA-T-011, PT-EXA-T-012, PT-EXA-T-013, PT-EXA-T-015		
<b>Tools Used</b>		<ul style="list-style-type: none"> <li></li> </ul>		
Step	Action	Expected Result	Status	Remarks
1	Access to the Visual Editor through the RAWFIE Web Portal	Redirection to the Visual Editor interface		
2	Open an already defined experiment	Experiment is presented in the editor		
3	Makes changes in the experiment workflow	The experiment is updated		
4	Save the experiment	The experiment is stored in the database and specific files are produced to be adopted by the remaining RAWFIE components		

**Table 19: Verification test of the Editor switching**

Test ID: <b>EAT05</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier – middle tier)</b>
<b>Hardware Configuration</b>		-		
<b>Software Configuration</b>		<ul style="list-style-type: none"> <li></li> </ul>		
<b>Test Name:</b>		<i>Switch between the Editors</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>User entered in the RAWFIE Portal</li> </ul>		
<b>Related Requirements</b>		PT-EXA-T-001, PT-EXA-T-002, PT-EXA-T-003, PT-EXA-T-004, PT-EXA-T-005, PT-EXA-T-008, PT-EXA-T-009, PT-EXA-T-010, PT-EXA-T-011, PT-EXA-T-012, PT-EXA-T-013, PT-EXA-T-015		
<b>Tools Used</b>		<ul style="list-style-type: none"> <li></li> </ul>		
Step	Action	Expected Result	Status	Remarks
1	Access to the editors through the RAWFIE Web Portal	Redirection to the editors interface		
2	Create an experiment	Experiment is presented in the editors		
3	Switch to the alternative editor and make changes	The experiment is updated		
4	Save the experiment	The experiment is stored in the database and specific files are produced to be adopted by the		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

		remaining RAWFIE components		
--	--	-----------------------------	--	--

**Table 20: Verification test of the experiment Launchings**

Test ID: <b>EAT05</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier – middle tier)</b>
<b>Hardware Configuration</b>		-		
<b>Software Configuration</b>		•		
<b>Test Name:</b>		<i>Launch experiments</i>		
<b>Preconditions</b>		• User entered in the RAWFIE Portal		
<b>Related Requirements</b>		PT-EXA-T-001, PT-EXA-T-002, PT-EXA-T-003, PT-EXA-T-004, PT-EXA-T-005, PT-EXA-T-008, PT-EXA-T-009, PT-EXA-T-010, PT-EXA-T-011, PT-EXA-T-012, PT-EXA-T-013, PT-EXA-T-015		
<b>Tools Used</b>		•		
Step	Action	Expected Result	Status	Remarks
1	Access to the authoring tool through the RAWFIE Web Portal	Redirection to the editors interface		
2	Select an experiment	A drop down list of the available experiments is appeared and the experimenter has the opportunity to select one		
3	Start the experiment execution	The launching service is informed with the experiment ID and the execution starts		

### 6.1.1.5 Experiment Monitoring Tool



**Table 21: Verification test of the Visualization of experiment status**

Test ID: EMT01		Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier)</b>
<b>Hardware Configuration</b>		-		
<b>Software Configuration</b>		-		
<b>Test Name:</b>		<i>Visualisation of experiment status</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Experiments running knowledge about the experiments state needed on user side (to check results)</li> </ul>		
<b>Related Requirements</b>		PT-EXM-T-001,		
<b>Tools Used</b>		<ul style="list-style-type: none"> <li></li> </ul>		
Step	Action	Expected Result	Status	Remarks
1	user opens Experiment Monitoring Tool in the Web Portal	Experiment Monitoring Tool displays a view with all experiments of the current user (ordered by date descending). The list also contains a sort summary of the experiments state		
2	user selects a experiment	Experiment Monitoring Tool displays all experiment details (date / timespan; related testbed; list of used UxVs; execution state ; link to the used EDL)		

**Table 22: Verification test of the canceling of experiments**

Test ID: EMT02		Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier)</b>
<b>Hardware Configuration</b>		-		
<b>Software Configuration</b>		-		
<b>Test Name:</b>		<i>Cancel of experiment</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Experiments running</li> </ul>		
<b>Related Requirements</b>		PT-EXP-C-001, PT-LAU-S-010, PT-LAU-S-012, TB-MAN-005		
<b>Tools Used</b>		<ul style="list-style-type: none"> <li></li> </ul>		
Step	Action	Expected Result	Status	Remarks
1	user opens Experiment Monitoring Tool in the Web Portal	Experiment Monitoring Tool displays a view with all experiments of the current user		
2	user selects a experiment	Experiment Monitoring Tool displays all experiment details and the option to cancel it		
3	User clicks the cancel button	Cancellation request is sent. User is informs the cancelation ongoing		
4	User watches further the experiment status	Experiments status is set to cancels when the cancelation has been performed		

6.1.1.6 System Monitoring Tool





## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 23: Verification test of the Visualisation of system and UxV health status**

Test ID: <b>SMT01</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Visualisation of system and UxV health status</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>connection to the System Monitoring Service (may not be necessary if System Monitoring Service collects all necessary data anyway)</li> <li>administrative knowledge about the system state needed on user side (to check results)</li> </ul>		
<b>Related Requirements</b>		PT-SYM-T-001		
<b>Tools Used</b>		<ul style="list-style-type: none"> <li></li> </ul>		
Step	Action	Expected Result	Status	Remarks
1	user opens System Monitoring Tool in the Web Portal	the System Monitoring Tool displays views with status of, middleware components, testbeds components, UxVs components		

### 6.1.1.7 UxV Navigation Tool

**Table 24: Verification test of the UxV navigation tool access and produced instructions validation**

Test ID: <b>UxVNT01</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier – middle tier)</b>
<b>Hardware Configuration</b>		-		
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Validate Experiments</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Requires Web Portal to be functioning and accessible</li> </ul>		
<b>Related Requirements</b>		PT-EXV-S-001, PT-EXV-S-002, PT-EXV-S-003		
<b>Tools Used</b>		<ul style="list-style-type: none"> <li></li> </ul>		
Step	Action	Expected Result	Status	Remarks
1	Access the UxV Navigation Tool through the portal	Ability to navigate the swarm		
2	Validate the produced instructions Validate the schema of the JSON output file Validate the data format of the JSON output file Validate the size of the JSON output file	All validation successful. The output data should be accessible and compatible with the required format		



6.1.1.8 Visualisation Tool

Table 25: Verification test of the User request handling

Test ID: VIS01		Conducted by:	Date:	Test Category: <b>Verification Tests (front end)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>User request handling</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Requires visualization tool to be functioning &amp; accessible.</li> <li>Requires visualization engine to be functioning &amp; accessible.</li> </ul>		
<b>Related Requirements</b>		PT-VIS-T-001		
<b>Tools Used</b>		<ul style="list-style-type: none"> <li></li> </ul>		
Step	Action	Expected Result	Status	Remarks
1	User sends a predefined websocket request via the visualization tool	The visualization tool forwards it to the visualization engine		
2	Handle the response from the visualization engine	The response is visualized on the user screen		

Table 26: Verification test of the Geospatial data handling

Test ID: VIS02		Conducted by:	Date:	Test Category: <b>Verification Tests (front end)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Geospatial data handling</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Requires visualization tool to be functioning &amp; accessible.</li> <li>Requires visualization engine to be functioning &amp; accessible.</li> <li>Requires message bus to be functioning &amp; accessible.</li> </ul>		
<b>Related Requirements</b>		PT-VIS-T-001, PT-VIS-T-002, PT-VIS-T-004, PT-VIS-T-005, PT-VIS-T-006, PT-VIS-T-007		
<b>Tools Used</b>		<ul style="list-style-type: none"> <li></li> </ul>		
Step	Action	Expected Result	Status	Remarks
1	Acquire predefined geospatial data (WMS, WFS) via the message bus	Data is properly received in the correct format at the VE		
2	Modify the data to be suited for the VT and send it via websocket to VT	VT renders the data and plots it on the screen		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 27: Verification test of the Geospatial data modification**

Test ID: <b>VIS03</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (front end)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Geospatial data modification</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Requires visualization tool to be functioning &amp; accessible.</li> <li>Requires visualization engine to be functioning &amp; accessible.</li> <li>Requires message bus to be functioning &amp; accessible.</li> </ul>		
<b>Related Requirements</b>		PT-VIS-T-001, PT-VIS-T-002, PT-VIS-T-004, PT-VIS-T-005, PT-VIS-T-006, PT-VIS-T-007		
<b>Tools Used</b>		<ul style="list-style-type: none"> <li>Browser</li> </ul>		
Step	Action	Expected Result	Status	Remarks
1	Acquire predefined geospatial data (WMS, WFS) via the message bus	Data is properly received in the correct format at the VE		
2	Add a layer of information data and send it to the VT	VT plots the data and the layer properly		

**Table 28: Verification test of the Experiment Controller communication**

Test ID: <b>VIS04</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (front end)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Experiment Controller communication</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Requires experiment controller to be functioning &amp; accessible.</li> <li>Requires visualization engine to be functioning &amp; accessible.</li> </ul>		
<b>Related Requirements</b>		PT-VIS-T-001		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Receive a message that the experiment has started from the Experiment Controller	The visualization tool starts the experiment		
2	Receive a message that the experiment has stopped from the Experiment Controller	The VT stops the experiment		



**Table 29: Verification test of the Visualization Tool Interaction**

Test ID: <b>VIS05</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (front end)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Visualization Tool Interaction</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Requires visualization tool to be functioning &amp; accessible.</li> <li>Requires visualization engine to be functioning &amp; accessible.</li> </ul>		
<b>Related Requirements</b>		PT-VIS-T-001, PT-VIS-T-002, PT-VIS-T-003, PT-VIS-T-004, PT-VIS-T-005, PT-VIS-T-006, PT-VIS-T-007		
<b>Tools Used</b>		<ul style="list-style-type: none"> <li></li> </ul>		
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1	Enable/Disable different features of the visualization tool (e.g. show/hide speed web widget)	The user sees the updated plot (show/hide speed web widget)		

**Table 30: Verification test of the Camera interaction**

Test ID: <b>VIS06</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (front end)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Camera interaction</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Requires visualization tool to be functioning &amp; accessible.</li> <li>Requires visualization engine to be functioning &amp; accessible.</li> <li>Requires Experiment controller to be functioning &amp; accessible.</li> </ul>		
<b>Related Requirements</b>		PT-VIS-T-001, PT-VIS-T-002, PT-VIS-T-003, PT-VIS-T-004, PT-VIS-T-005, PT-VIS-T-006, PT-VIS-T-007		
<b>Tools Used</b>		<ul style="list-style-type: none"> <li></li> </ul>		
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1	Retrieve with the visualization engine quasi real time data from one UxV, processes it and send it to the visualization tool	The VT plots the data properly		
2	Change the camera view for the scenario	Data camera is adjusted		

6.1.1.9 Data Analysis Tool

**Table 31: Verification test of the provision of an interface to the Analysis Engine by the Analysis Tool**

Test ID: <b>PT-DAA-T-001</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier)</b>
<b>Hardware Configuration</b>		<ul style="list-style-type: none"> <li></li> </ul>		
<b>Software Configuration</b>		<ul style="list-style-type: none"> <li></li> </ul>		
<b>Test Name:</b>		<i>Analysis Tool will provide an interface to the Analysis Engine (DAE)</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Working message bus</li> </ul>		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

	<ul style="list-style-type: none"> <li>Working schema registry</li> <li>Working Data Analysis Tool</li> </ul>			
<b>Related Requirements</b>	PT-DAA-T-002, PT-DAA-T-001, PT-DAA-T-004, PT-DAA-T-005			
<b>Tools Used</b>				
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1	User logs in to the web portal	Login successful		
2	DAT queries available schemas from Schema Registry	All schemas are returned successfully		
3	DAT allows user to select the data they want to work with as well as the machine learning algorithm and hyper-parameters	Job is sent via message bus to the DAE		

**Table 32: Verification test of the ability of the Analysis Tool to query available data schemas**

Test ID: <b>PT-DAA-T-002</b>	Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier)</b>	
<b>Hardware Configuration</b>	•			
<b>Software Configuration</b>	•			
<b>Test Name:</b>	<i>Analysis Tool will be able to query available data schemas</i>			
<b>Preconditions</b>	<ul style="list-style-type: none"> <li>Working message bus</li> <li>Working schema registry</li> <li>Working Data Analysis Tool</li> </ul>			
<b>Related Requirements</b>	PT-DAA-T-003			
<b>Tools Used</b>				
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1	User logs in to the web portal	Login successful		
2	DAT queries available schemas from Schema Registry	All schemas are returned successfully		

**Table 33: Verification test of the ability of the Analysis Tool to read results from the results database**

Test ID: <b>PT-DAA-T-003</b>	Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier)</b>	
<b>Hardware Configuration</b>	•			
<b>Software Configuration</b>	•			
<b>Test Name:</b>	<i>Analysis Tool will be able to read results from the results database</i>			
<b>Preconditions</b>	<ul style="list-style-type: none"> <li>Working message bus</li> <li>Working schema registry</li> <li>Working Data Analysis Tool</li> <li>Working results database [graphite]</li> </ul>			
<b>Related Requirements</b>	PT-DAA-T-001, PT-DAA-T-005			
<b>Tools Used</b>				
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1	User logs in to the web portal	Login successful		
2	User builds job	Job successfully built (or error) and sent to DAE		



3	Results are shown in results tab	Job results are shown as they are processed via graphite UI		
---	----------------------------------	---	--	--

## 6.1.2 Middle Tier (Services and Communication components)

### 6.1.2.1 Testbed Directory Service

Table 34: Verification test of the resource Retrieval from testbed facility

Test ID: <b>TD01</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (Middle Tier)</b>	
<b>Hardware Configuration</b>					
<b>Software Configuration</b>					
<b>Test Name:</b>		<i>Retrieve resources information from testbed facility</i>			
<b>Preconditions</b>		Access to the PostgreSQL server must be granted for the Testbed Directory Service When preparing the test, the test executor should know either the ID of the testbed he is looking for (in case of the getResources() interface and when the list of all resources of the given testbed is required), or the list criteria for selecting particular resources (in case of the searchResourceAPI() method)			
<b>Related Requirements</b>		PT-DIR-S-003, PT-DIR-S-004, PT-DIR-S-006			
<b>Tools Used</b>		SOAP UI			
Step	Action	Expected Result	Status	Remarks	
1.a	The input JSON request is prepared, specifying the testbed identifier	No error occurred. The Testbed Directory Service gives back a JSON response message, containing details about all resources belonging to the specified testbed			
2.a	The getResources() REST interface is called from the SOAP UI Client Tool, providing the prepared JSON request in input				
1.	The input JSON request is prepared, specifying the testbed identifier, the identifier of the resource and / or specific resources parameters / characteristics	No error occurred. The Testbed Directory Service gives back a JSON response message, containing detailed information about the resources matching the search criteria and belonging to the specified testbed			
2.b	The searchResource() REST interface is called from the SOAP UI Client Tool, providing the prepared JSON request in input				



Table 35: Verification test of the Addition of a new testbed facility to the RAWFIE federation

Test ID: <b>TD02</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (Middle Tier)</b>	
<b>Hardware Configuration</b>					
<b>Software Configuration</b>					
<b>Test Name:</b>		<i>Add / delete a testbed facility to the RAWFIE federation</i>			
<b>Preconditions</b>		<p>Access to the PostgreSQL server must be granted for the Testbed Directory Service</p> <p>When preparing the test for the testbed registration case, the test executor should know the information e about the testbed to be inserted, according to what required by the platform for the registration process. Additionally, in case of a testbed deletion, the testbed id must be known in advance</p>			
<b>Related Requirements</b>		PT-DIR-S-005			
<b>Tools Used</b>		SOAP UI			
Step	Action	Expected Result	Status	Remarks	
1.a	The input JSON request is prepared, with the information about the new testbed to be added	No error occurred. And the information about the new testbed is from now on available in the Master Data Repository, as it can be verified by using the <code>getTestbeds()</code> or <code>searchTestbed()</code> REST interfaces (see <b>TD04</b> in the following)			
2.a	The <code>createTestbed()</code> REST interface is called from the SOAP UI Client Tool, specifying the testbed information in the input JSON request				
1.b	The input JSON message request is prepared, with the unique id of the testbed facility to be deleted	No error occurred. And the information about the deleted testbed (and related resources) is not available anymore in the Master Data Repository, as it can be verified by using the <code>getTestbeds()</code> or <code>searchTestbed()</code> REST interfaces (see <b>TD04</b> in the following)			
2.b	The <code>deleteTestbed()</code> REST interface is called from the SOAP UI Client Tool, specifying the information about the testbed to be deleted in the provided input JSON request				



**Table 36: Verification test of the Registration of a new UxV node into a testbed facility**

Test ID: <b>TD03</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (Middle Tier)</b>	
<b>Hardware Configuration</b>					
<b>Software Configuration</b>					
<b>Test Name:</b>		<i>Register / delete an UxV node into a testbed facility</i>			
<b>Preconditions</b>		<p>Access to the PostgreSQL server must be granted for the Testbed Directory Service</p> <p>When preparing the test in the case of the registration of a new resource, the test executor should know all information related to new resource to be added and the related testbed, according to the information required by the platform for the registration process. For the resource deletion, the testbed id and resource id should be known.</p>			
<b>Related Requirements</b>		PT-DIR-S-007			
<b>Tools Used</b>		SOAP UI			
Step	Action	Expected Result	Status	Remarks	
1.a	The input JSON message request is prepared, with all information about the new resource to be added (and the unique id of the testbed facility it belongs to)	No error occurred. And the information about the new resource (UxV node) is from now on available in the Master Data Repository, as it can be verified by using the getResources() or searchResource() REST API (see previous tests)			
2.a	The createResource() REST interface is called from the SOAP UI Client Tool, specifying the information about the resource to be added in the provided input JSON request				
1.b	The input JSON message request is prepared, with the unique id of the resource to be deleted and of the testbed facility it belongs to	No error occurred. And the resource (UxV node) is not available anymore in the Master Data Repository, as it can be verified by using the getResources() or searchResource() REST API (see previous tests)			
2.b	The deleteResource() REST interface is called from the SOAP UI Client Tool, specifying the information about the resource to be deleted in the provided input JSON request				





## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 37: Verification test of the Retrieval of testbed information and belonging resources**

Test ID: <b>TD04</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (Middle Tier)</b>	
<b>Hardware Configuration</b>					
<b>Software Configuration</b>					
<b>Test Name:</b>		<i>Retrieve testbed information and belonging resources</i>			
<b>Preconditions</b>		<p>Access to the PostgreSQL server must be granted for the Testbed Directory Service</p> <p>When preparing the test, the test executor should know the ID of the testbed he is looking for, in case only information of resources form a very specific testbed is required. Or it can just provide one or a list of search criteria</p>			
<b>Related Requirements</b>		PT-DIR-S-001, PT-DIR-S-002, PT-DIR-S-006			
<b>Tools Used</b>					
Step	Action	Expected Result	Status	Remarks	
1.a	The getTestbeds() REST interface is called from the SOAP UI Client Tool, without any specific testbed information (null JSON input request)	No error occurred. The Testbed Directory Service gives back a JSON response message, containing details about all registered testbeds and all resources belonging to each of them			
1.b	An input JSON message with the information about the identifier of the testbed we are requesting information about, is prepared	No error occurred. The Testbed Directory Service gives back a JSON response message, containing details about the available testbeds conforming to the search criteria			
2.b	The getTestbeds() REST interface is called from the SOAP UI Client Tool, using the abovementioned JSON as input message request	No error occurred. The Testbed Directory Service gives back a JSON response message, containing details about the available testbeds conforming to the search criteria			
1.c	An input JSON message is prepared, specifying the filters / list of criteria to search for testbeds with specific characteristics, e.g.: testbed id, UxV types, testbed technological capabilities, as keywords, and so on	No error occurred. The Testbed Directory Service gives back a JSON response message, containing details about the available testbeds conforming to the search criteria			
2.c	The searchTestbed() REST interface is called from the SOAP UI Client Tool, using the abovementioned JSON as input message request	No error occurred. The Testbed Directory Service gives back a JSON response message, containing details about the available testbeds conforming to the search criteria			



6.1.2.2 EDL Compiler and Validator

Table 38: Verification test of the Experiments compilation

Test ID: ECV01		Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier – middle tier)</b>
<b>Hardware Configuration</b>		-		
<b>Software Configuration</b>		•		
<b>Test Name:</b>		<i>Compile Experiments</i>		
<b>Preconditions</b>		• User entered in the RAWFIE Portal		
<b>Related Requirements</b>		PT-CPV-001, PT-CPV-002, PT-CPV-003, PT-CPV-004, PT-EXV-S-001, PT-EXV-S-002, PT-EXV-S-003		
<b>Tools Used</b>				
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1	Access to the authoring tool through the RAWFIE Web Portal	Redirection to the editors interface		
2	Write a simple experiment	The experiment workflow is presented in the available editors		
3	Compile the experiment	The necessary files required by the remaining RAWFIE components are produced		

Table 39: Verification test of the Experiments validation

Test ID: ECV02		Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier – middle tier)</b>
<b>Hardware Configuration</b>		-		
<b>Software Configuration</b>		•		
<b>Test Name:</b>		<i>Validate Experiments</i>		
<b>Preconditions</b>		• User entered in the RAWFIE Portal		
<b>Related Requirements</b>		PT-CPV-001, PT-CPV-002, PT-CPV-003, PT-CPV-004, PT-EXV-S-001, PT-EXV-S-002, PT-EXV-S-003		
<b>Tools Used</b>		•		
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1	Access to the authoring tool through the RAWFIE Web Portal	Redirection to the editors interface		
2	Write a simple experiment	The experiment workflow is presented in the available editors		
3	Validate the experiment	Validation is performed and error / warning messages are presented in the editors		



6.1.2.3 Users & Rights Service

Table 40: Verification test of the Users & Rights Service login checking

Test ID: URS01		Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Login checking</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Valid user name and password known</li> </ul>		
<b>Related Requirements</b>		<ul style="list-style-type: none"> <li>PT-USR-S-001</li> </ul>		
<b>Tools Used</b>		<ul style="list-style-type: none"> <li></li> </ul>		
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1	invalid user name and password sent to the Users & Rights Service	Users & Rights Service returns failure		
2	valid user name and password sent to the Users & Rights Service	Users & Rights Service returns OK		

Table 41: Verification test of the Users & Rights Service roles/rights checking

Test ID: URS02		Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Roles/rights checking</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>User with the tested roles is available</li> </ul>		
<b>Related Requirements</b>		PT-USR-S-002		
<b>Tools Used</b>		<ul style="list-style-type: none"> <li></li> </ul>		
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1	Role request with not available roles for a user is sent to the Users & Rights Service	Users & Rights Service returns failure		
2	Role request with available roles for a user is sent to the Users & Rights Service	Users & Rights Service returns OK		



Table 42: Verification test of the user rights checks

Test ID: <b>URS03</b>	Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>	
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>	<i>Check user rights</i>			
<b>Preconditions</b>	<ul style="list-style-type: none"> <li>Valid user rights known</li> </ul>			
<b>Related Requirements</b>	PT-USR-S-001, PT-USR-S-002, PT-USR-S-003			
<b>Tools Used</b>	<ul style="list-style-type: none"> <li>SOAPUI REST client</li> </ul>			
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1	user ID and available required rights sent to the Users & Rights Service	Users & Rights Service return true	Success	
2	user ID and not available required rights sent to the Users & Rights Service	Users & Rights Service return false	Success	

6.1.2.4 Booking Service

Booking Service main role is:

- to check the Master DB, ensure no conflicts with other reservation exists
- performs all necessary updates/inserts/deletions related to editing or creation of reservations
- informs the involved users (creator and testbed operator) by sending appropriate notifications (emails) regarding reservation status changes

Booking Service is tightly coupled with the Booking Tool component therefore, the verification tests described in section 6.1.1.3 should also be considered during Booking Service functionality verification activities. Verification tests of the component focus around testing and ensuring the correctness of each provided method. The class diagram provided in D4.5 section 4.2.6 – Booking Service can form the source for defining the actual verification tests.

The Booking Service requirements not addressed by the tests specified below are PT-BOO-S-003, PT-BOO-S-009 and PT-BOO-S-010 .



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 43: Verification test of Booking Service add reservation functionality**

Test ID: <b>BS01</b>	Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>	
<b>Hardware Configuration</b>	-			
<b>Software Configuration</b>	-			
<b>Test Name:</b>	<i>Booking Service add reservation functionality</i>			
<b>Preconditions</b>	<ul style="list-style-type: none"> <li>Master DB is prepopulated with reservations of different status and timeslots (involved tables are: Reservation, Resource Reservation)</li> <li>User initiating the call is a valid experimenter</li> </ul>			
<b>Related Requirements</b>	PT-BOO-S-001 (experiment level booking) PT-BOO-S-002 PT-BOO-S-004 PT-BOO-S-005 PT-BOO-S-007 PT-BOO-S-011			
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Call addReservation() providing a datetime interval that has passed	response should be returned with a proper failure message		
2	Call addReservation() providing a datetime interval in the future (NO conflict in requested resources with existing reservation at the same time)	Appropriate MasterDB tables are updated (new reservation in status=PENDING)		
		If email sending is enabled then email is send to both the creator and the testbed operator of the reserved resources		
		The returned response contains the newly created reservationId and the reservation status		
3	Call addReservation() providing a datetime interval in the future conflict in requested resources with existing reservation at the same time)	response should be returned with a proper failure message		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 44: Verification test of Booking Service edit reservation functionality**

Test ID: <b>BS02</b>	Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>	
<b>Hardware Configuration</b>	-			
<b>Software Configuration</b>	-			
<b>Test Name:</b>	<i>Booking Service add reservation functionality</i>			
<b>Preconditions</b>	<ul style="list-style-type: none"> <li>Master DB is prepopulated with reservations of different status and timeslots (involved tables are: Reservation, Resource_Reservation)</li> <li>User initiating the call is a valid experimenter</li> </ul>			
<b>Related Requirements</b>	PT-BOO-S-002 PT-BOO-S-005 PT-BOO-S-007			
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Call editReservation() providing appropriate ReservationData which should include the reservationId (the call should include credentials about the user initiating it)	If provided user credentials do not match with the ones of the reservation owner then a proper failure message is returned		
		If existing reservation status!= PENDING then no update should be possible and a proper failure message is returned		
		If time related changes refer to an interval in the past then a proper failure message is returned		
	(If status= PENDING & user credential match)	If overlaps with existing reservation are introduced and resources conflicts are detected then a proper failure message is returned		
	(If status= PENDING & user credential match)	If no resources conflicts are detected the changes are accepted and the corresponding DB tables updated		
2	Repeat step 1 with different kind of changes related to timeslots and resource selection	Ensure that expected results are respected as described in step 1		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 45: Verification test of Booking Service approve reservation functionality**

Test ID: <b>BS03</b>	Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>	
<b>Hardware Configuration</b>	-			
<b>Software Configuration</b>	-			
<b>Test Name:</b>	<i>Booking Service approve reservation functionality</i>			
<b>Preconditions</b>	<ul style="list-style-type: none"> <li>Master DB is prepopulated with reservations of different status and timeslots (involved tables are: Reservation, Resource_Reservation)</li> </ul>			
<b>Related Requirements</b>	PT-BOO-S-002 PT-BOO-S-005 PT-BOO-S-007 PT-BOO-S-011 PT-NF-002			
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Call approveReservation() (the call should include credentials about the user initiating it)	If provided credentials do not match with an authorized platform user then a proper failure message is returned		
		If provided credentials do not refer to an authorized platform user with role=TESTBED_OP then a proper failure message is returned		
		If reservationId refers to a reservation with status !=PENDING then a proper failure message is returned		
		If reservationId refers to a past reservation then then a proper failure message is returned		
		If conflicts are detected with any other APPROVED reservation then then a proper failure message is returned		
2	(If status= PENDING & caller=TESTBED_OP & no conflicts detected	Status change is accepted and corresponding DB tables updated		
		An email is send to the owner of the reservation		
		A ReservationStatusMsg is send to Message bus		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 46: Verification test of Booking Service reject reservation functionality**

Test ID: <b>BS04</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>	
<b>Hardware Configuration</b>		-			
<b>Software Configuration</b>		-			
<b>Test Name:</b>		<i>Booking Service reject reservation functionality</i>			
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Master DB is prepopulated with reservations of different status and timeslots (involved tables are: Reservation, Resource_Reservation)</li> </ul>			
<b>Related Requirements</b>		PT-BOO-S-002 PT-BOO-S-005 PT-BOO-S-007 PT-BOO-S-011 PT-NF-002			
<b>Tools Used</b>					
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>	
1	Call approveReservation() (the call should include credentials about the user initiating it)	If provided credentials do not match with an authorized platform user then a proper failure message is returned			
		If provided credentials do not refer to an authorized platform user with role=TESTBED_OP then a proper failure message is returned			
		If reservationId refers to a reservation with status !=PENDING or APPROVED then a proper failure message is returned			
		If reservationId refers to a past reservation then then a proper failure message is returned			
2	(If status= PENDING & caller=TESTBED_OP	Status change is accepted and corresponding DB tables updated			
		An email is send to the owner of the reservation			
		A ReservationStatusMsg is send to Message bus			





## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 47: Verification test of Booking Service delete reservation functionality**

Test ID: <b>BS05</b>	Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>	
<b>Hardware Configuration</b>	-			
<b>Software Configuration</b>	-			
<b>Test Name:</b>	<i>Booking Service delete reservation functionality</i>			
<b>Preconditions</b>	<ul style="list-style-type: none"> <li>Master DB is prepopulated with reservations of different status and timeslots (involved tables are: Reservation, Resource_Reservation)</li> </ul>			
<b>Related Requirements</b>	PT-BOO-S-002 PT-BOO-S-005 PT-BOO-S-007 PT-NF-002			
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Call deleteReservation() (the call should include credentials about the user initiating it)	If provided credentials do not match with an authorized platform user then a proper failure message is returned		
		If reservationId refers to a past reservation then a proper failure message is returned		
		If reservationId refers to a reservation with resources involved in a currently running experiment a proper failure message is returned		
		If none of the above then status change to CANCELLED		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 48: Verification test of Booking Service retrieve reservation(s) functionality**

Test ID: <b>BS06</b>	Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>	
<b>Hardware Configuration</b>	-			
<b>Software Configuration</b>	-			
<b>Test Name:</b>	<i>Booking Service retrieve reservation(s) functionality</i>			
<b>Preconditions</b>	<ul style="list-style-type: none"> <li>Master DB is prepopulated with reservations of different status and timeslots (involved tables are: Reservation, Resource_Reservation)</li> </ul>			
<b>Related Requirements</b>	PT-BOO-S-002 PT-BOO-S-008			
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Call getReservation() providing a reservationId	Inspect response and ensure data is inline with the information stored in the MasterDB		
2	Call getReservations() providing appropriate search criteria (time, user etc.)	Inspect response and ensure data is in line with the information stored in the MasterDB		

**Table 49: Verification test of Booking Service check for conflicts functionality**

Test ID: <b>BS07</b>	Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>	
<b>Hardware Configuration</b>	-			
<b>Software Configuration</b>	-			
<b>Test Name:</b>	<i>Booking Service check for conflicts functionality</i>			
<b>Preconditions</b>	<ul style="list-style-type: none"> <li>Master DB is prepopulated with reservations of different status and timeslots (involved tables are: Reservation, Resource_Reservation)</li> </ul>			
<b>Related Requirements</b>	PT-BOO-S-002 PT-BOO-S-008			
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Call checkForConflictingReservations() providing proper reservation data info	Returns true or false depending on whether resource conflicts are detected for time overlapping with pre-existing in the MasterDB reservations		
2	Call getReservations() providing appropriate search criteria (time, user etc.)	Inspect response and ensure data is in line with the information stored in the MasterDB		



**Table 50: Verification test of Booking Service simultaneous reservations support**

Test ID: <b>BS08</b>	Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>	
<b>Hardware Configuration</b>	-			
<b>Software Configuration</b>	-			
<b>Test Name:</b>	<i>Booking Service simultaneous reservations support</i>			
<b>Preconditions</b>	<ul style="list-style-type: none"> <li>Master DB is prepopulated with reservations of different status and timeslots (involved tables are: Reservation, Resource_Reservation)</li> </ul>			
<b>Related Requirements</b>	PT-BOO-S-002 PT-BOO-S-009			
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Multiple calls of Booking Service addReservation() method (execute BS01 multiple times simultaneously from different clients)	Ensure that all requests are processed and multiple reservations are created in the MasterDB		

6.1.2.5 *Launching Service*

The Launching Service requirements, which are not addressed by the tests specified below, are:

- PT-BOO-S-011



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 51: Verification test of the Launching Service manualStart (short term launching)**

Test ID: <b>LS01</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Experiment short term launching</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Requires the Message Bus and the experiment controller to be accessible.</li> <li>The master data repository should contain reservations for the user and for a defined experiment (involved tables are Experiment Experiment_Execution., Reservation, Reservation_item)</li> </ul>		
<b>Related Requirements</b>		PT-LAU-S-001 PT-LAU-S-003 PT-LAU-S-004 PT-LAU-S-005 PT-LAU-S-007 PT-LAU-S-008 PT-LAU-S-009 (by design) PT-LAU-S-012 PT-LAU-S-013 (by design)		
<b>Tools Used</b>				
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1	User call manualStart() providing an experiment Id	if experimentId is not present in the MasterDB then a proper failure message is returned		
		If supplied user credentials do not match an authorized user then a proper failure message is returned		
		If supplied user credentials match an authorized user but refer to booked resources of another user then a proper failure message is returned		
2	(case experimentId exists)	if an executionId already exists and refers to a running experiment (status=Ongoing) then a proper failure message is returned		
3	(case no executionId exists or exists for an status!=Ongoing)	Launching service generates an ExperimentStartRequest to the Message Bus (targeting the Experiment Controller).		
		Master DB tables are properly updated (tables Experiment_Execution, Reservation_item)		
		LaunchingServiceActionResp json message is returned containing the generated executionId and the status of the experiment		



D4.6 - Pilot Experimentation Scenarios for Validation and Testing

Table 52: Verification test of the Launching Service schedule (long term launching)

Test ID: <b>LS02</b>	Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>	
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b> <i>Experiment long term launching</i>				
<b>Preconditions</b>				
<ul style="list-style-type: none"> <li>Requires the Message Bus and the experiment controller to be accessible.</li> <li>The master data repository should contain reservations for the user and for a defined experiment (involved tables are Experiment Experiment_Execution., Reservation, Reservation_item)</li> <li>The platform launching scheduler must be running</li> </ul>				
<b>Related Requirements</b>				
PT-LAU-S-002 PT-LAU-S-003 PT-LAU-S-004 PT-LAU-S-005 PT-LAU-S-007 PT-LAU-S-008 PT-LAU-S-009 (by design) PT-LAU-S-012 PT-LAU-S-013 (by design) PT-BOO-S-011				
<b>Tools Used</b>				
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1	User call schedule() providing experimentId, startDate, endDate	if experimentId is not present in the MasterDB then a proper failure message is returned		
		If supplied user credentials do not match an authorized user then a proper failure message is returned		
		If supplied user credentials match an authorized user but refer to booked resources of another user then a proper failure message is returned		
		If startDate or, endDate refer to past time then a proper failure message is returned		
		If startDate or endDate are not contained within the timeslot defined for the associated reservation then a proper failure message is returned		
		if an executionId already exists and refers to a running experiment (status=Ongoing) then a proper failure message is returned		
2	Scheduling part (case all preconditions are met)	Launching Scheduler is called and a job is added to be launched at the specified startDate		
		The user (owner) of the experiment and the testbed operator are informed by an appropriate notification (email)		
		Master DB tables are properly updated (tables Experiment_Execution, Reservation_item). The status of the		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

		experiment should be BOOKED		
		LaunchingServiceActionResp json message is returned containing the generated executionId and the status of the experiment		
3	Execution part (check Launching Service activity when startDate arrives)	<p>Master DB tables are properly updated (tables Experiment_Execution, Reservation_item)</p> <p>The status of the experiment changes to ONGOING</p>		
		Launching service generates an ExperimentStartRequest to the Message Bus (targeting the Experiment Controller).		
		Scheduled job (for the executionId) is removed from scheduler		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 53: Verification test of the Launching Service cancellation request**

Test ID: <b>LS03</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Experiment cancellation request</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Requires the Message Bus and the experiment controller to be accessible.</li> <li>The master data repository should contain reservations for the user and for a defined experiment (involved tables are Experiment Execution, Reservation, Reservation_item)</li> <li>An experiment should be schedule for a future time</li> </ul>		
<b>Related Requirements</b>		PT-LAU-S-009 (by design) PT-LAU-S-010 PT-LAU-S-012 PT-LAU-S-013 (by design)		
<b>Tools Used</b>				
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1	User call cancellation() providing an executionId	if executionId is not present in the MasterDB then a proper failure message is returned		
		If supplied user credentials do not match an authorized user then a proper failure message is returned		
		If supplied user credentials match an authorized user but refer to an experiment of another experimenter then a proper failure message is returned (Exception to this rule if credentials refer to a testbed operator or administrator)		
2	(case executionId exists)	If the experiment is already running (status= ONGOING) then cancellation is not possible and a proper failure message is returned		
		If no schedule job is found in Launching scheduler then a proper failure message is returned		
3	(executionId exists and the execution is still in the scheduler)	Job is removed from the scheduler		
		Master DB tables are properly updated (tables Experiment_Execution, Reservation_item)  The status of the experiment changes to CANCELLED		
		LaunchingServiceActionResp json message is returned containing with the executionId,		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

		status= CANCELLED and empty message field		
		The user (owner) of the experiment and the testbed operator are informed by an appropriate notification (email)		

**Table 54: Verification test of Launching Service simultaneous launching capability**

Test ID: <b>LS04</b>	Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>	
<b>Hardware Configuration</b>	-			
<b>Software Configuration</b>	-			
<b>Test Name:</b>	<i>Launching Service simultaneous launching capability</i>			
<b>Preconditions</b>	<ul style="list-style-type: none"> <li>Master DB is prepopulated with reservations of different status and timeslots (involved tables are: Reservation, Resource_Reservation)</li> </ul>			
<b>Related Requirements</b>	PT-LAU-S-006			
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Multiple calls of Launching Service schedule() method (execute LS01 multiple times simultaneously from different clients)	Ensure that all requests are processed multiple experiments executions exist in the Job Scheduler		

### 6.1.2.6 Visualisation Engine

**Table 55: Visualisation engine user request handling**

Test ID: <b>VE01</b>	Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>	
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>	<i>Connection Test</i>			
<b>Preconditions</b>	<ul style="list-style-type: none"> <li>Requires visualization tool and visualization engine to function and be accessible</li> </ul>			
<b>Related Requirements</b>	VIS01			
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Visualization engine receive through websocket request from visualization tool	The visualization engine handles the request		
2	Visualization engine sends through websocket the response	Visualization tool receives response		





## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 56: Visualisation engine user request handling**

Test ID: VE02		Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Connection Test</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Requires visualization tool and visualization engine to function and be accessible</li> </ul>		
<b>Related Requirements</b>		VIS01, VIS02		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Visualization engine receive through websocket request from visualization tool	The visualization engine handles the request		
2	Visualization engine sends through websocket the response	Visualization tool receives response		

**Table 57: Visualization engine geospatial data modification**

Test ID: VE03		Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Connection Test</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Requires visualization tool and visualization engine to function and be accessible</li> </ul>		
<b>Related Requirements</b>		VIS01, VIS02		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Visualization engine receive through the message bus	The visualization engine handles the request		
2	Visualization engine update data in database	Data is properly stored in the database for future retrieval		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 58: Visualization engine camera interaction**

Test ID: <b>VE04</b>	Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>	
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>	<i>Connection Test</i>			
<b>Preconditions</b>	<ul style="list-style-type: none"> <li>Requires visualization tool and visualization engine to function and be accessible and the UxV to send video data</li> </ul>			
<b>Related Requirements</b>	VIS01			
<b>Tools Used</b>				
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1	Visualization engine receive request from visualization tool to start the camera stream	Visualization engine forward this request to the UxV		

### 6.1.2.7 Data Analysis Engine

**Table 59: Verification test of the ability of the Analysis Engine to query message bus streams & schemas from the schema registry**

Test ID: <b>PT-DAA-E-001</b>	Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier)</b>	
<b>Hardware Configuration</b>	<ul style="list-style-type: none"> <li></li> </ul>			
<b>Software Configuration</b>	<ul style="list-style-type: none"> <li>Spark 1.6</li> <li>Graphite 0.9</li> <li>Confluent 2.01</li> </ul>			
<b>Test Name:</b>	<i>Analysis Engine will be able to query message bus streams &amp; schemas from the schema registry</i>			
<b>Preconditions</b>	<ul style="list-style-type: none"> <li>Working message bus</li> <li>Working schema registry</li> <li>Working Data Analysis Tool</li> </ul>			
<b>Related Requirements</b>	PT-DAA-S -001, PT-DAA-S -002			
<b>Tools Used</b>				
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1	User deploys job (currently via CLI, but in the future via web UI)	DAE checks if job is a pre-existing jar, else compiles a new one		
2	DAE verifies schema from registry and starts a spark job that acquires data from the message bus	The job is successfully build and uploaded to the job server		

**Table 60: Verification test of the ability of the Analysis Engine to receive messages from the Analysis Tool**

Test ID: <b>PT-DAA-E-002</b>	Conducted by:	Date:	Test Category: <b>Verification Tests (front end tier)</b>
------------------------------	---------------	-------	---



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

<b>Hardware Configuration</b>		•		
<b>Software Configuration</b>		•		
<b>Test Name:</b>		<i>Analysis Engine will be able to receive messages from the Analysis Tool</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Working message bus</li> <li>Working schema registry</li> <li>Working Data Analysis Tool</li> </ul>		
<b>Related Requirements</b>		PT-DAE-001 (PT-DIR-S-001)		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	User builds a job on the Data Analysis Tool	Job is successfully checked for errors		
2	Data Analysis Engine receives job via message bus and builds a job	The job is successfully compiled (or an error returned)		
3	Data Analysis Engine builds job and sends data to Spark	The job is converted to a JAR and uploaded via REST to the Spark job server		

**Table 61: Verification test of the ability of the Analysis Engine to write data to the results database**

<b>Test ID: PT-DAA-E-003</b>		<b>Conducted by:</b>	<b>Date:</b>	<b>Test Category: Verification Tests (front end tier)</b>
<b>Hardware Configuration</b>		•		
<b>Software Configuration</b>		•		
<b>Test Name:</b>		<i>Analysis Engine will be able to write data to the results database</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Working message bus</li> <li>Working schema registry</li> <li>Working Data Analysis Engine</li> <li>Working Graphite Instance</li> </ul>		
<b>Related Requirements</b>		PT-DIR-S-002		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	User builds a job and the jar is uploaded to the spark job server	Job is uploaded successfully and the job server registers the job in spark		
2	Spark Engine sends results to the Graphite instance as it processes the data	Graphite displays a runtime stream of processed data		



6.1.2.8 System Monitoring Service

Table 62: Verification test of the System Monitoring

Test ID: SYMS01		Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>System Monitoring</i>		
<b>Preconditions</b>		•		
<b>Related Requirements</b>		PT-SYM-S-001, PT-SYM-S-002		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Service polls the computes of the middle tier for their status	Computes return their health status to the service		
2	Service listen to status messages on the message bus	Testbed component sent automatically status information on the message bus. Messages received by the service		
3	System Monitory Tool request status information	Service collects the information and returns it		

Table 63: Verification test of the System Monitoring Problem Notifications

Test ID: SYMS02		Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>System Monitoring Problem Notifications</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>• Notification receivers are configured</li> <li>• Status information is collected</li> </ul>		
<b>Related Requirements</b>		PT-SYM-S-003		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Problem occurred (serer down etc.)	Services send email notifications of the configured receivers.		
2	System Monitory Tool request status information	Problems are visualized in the System Monitory Tool		



6.1.2.9 Accounting Service

Table 64: Verification test of the Accounting data collection

Test ID: ACCS01		Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Accounting data collection</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Accounting data is empty for the used user</li> </ul>		
<b>Related Requirements</b>		PT-ACC-S-002, PT-ACC-S-003		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Experiment is completed. Notifications sent on the message bus.	Accounting received the event and computes the charge for the experiments		
2	Billing period ends	Bill is sent to the user		

6.1.2.10 Experiment Controller

Table 65: Verification test of Experiment Controller connection

Test ID: EC01		Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Connection Test</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Requires web portal to be functioning and accessible.</li> <li>Register an experiment (Testbed manager)</li> <li>Send Network Requirements (Testbed manager)</li> <li>Send basic instructions to the Resource Controller</li> <li>Transmit simulated or real results back to the Experiment Monitoring Tool</li> </ul>		
<b>Related Requirements</b>		TB-NEC-004		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Register an experiment (Testbed manager)	Successful registration		
2	Send Network Requirements (Testbed manager)	Network requirements met, acknowledged by the Testbed Controller		
3	Send basic instructions to the Resource Controller	Instructions acknowledged by the Resource Manager (resources are available)		
4	Transmit simulated or real results back to the Experiment Monitoring Tool	Results successfully received by the Experiment Monitoring Tool		



**Table 66: Verification test of Experiment Controller workflow**

Test ID: <b>EC02</b>	Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>	
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>	<i>Execute experiment workflow</i>			
<b>Preconditions</b>	<ul style="list-style-type: none"> <li>The experimenter has already created the script for the experiment of interest</li> <li>The chosen resource must be completely available and ready to use</li> </ul>			
<b>Related Requirements</b>	PT-EXP-C-001, PT-EXP-C-002, PT-EXP-C-003, PT-EXP-C-004, PT-EXP-C-005, PT-EXP-C-006, PT-EXP-C-007			
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	The experimenter forwards the script to the Experiment Controller in order to start or barely execute the next action of the resource mission	Successful forwarding and start of execution		
2	The instructions are forwarded to the corresponding testbed facility	Testbed facility received the instructions correctly		
3	The resource receives the new set of instructions as generated from the script for overriding the experiment workflow	The resource overrides its current experiment according to the new instructions		

### 6.1.3 Testbed Tier (Testbeds and Resources control components)

#### 6.1.3.1 Monitoring Manager



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 67: Verification test of Monitoring Activity**

Test ID: <b>MM01</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Check Monitoring Activity</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Requires the resource controller to be accessible.</li> <li>Requires the network controller to be accessible.</li> <li>Requires the data tier to be accessible.</li> </ul>		
<b>Related Requirements</b>		PT-SYM-T01, TB-MOM-001, TB-MOM02, TB-MOM-003, TB-MOM-004		
<b>Tools Used</b>				
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1	The Monitoring Manager ‘checks’ the status of the resources through the Resource Controller.	The Resource Controller informs the Monitoring Manager for malfunctions of the status of UxVs		
2	Monitoring Manager periodically forwards the messages to the message bus	Topics about the UxVs system status are updated by Monitoring Manager		

### 6.1.3.2 Network Controller

**Table 68: Verification test of network interface switching due to connectivity problems**

Test ID: <b>NC01</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Switch network interface due to connectivity problem</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Requires the Testbed Manager to be accessible</li> </ul>		
<b>Related Requirements</b>		TB-NEC-001, TB-NEC-003, TB-NEC-004		
<b>Tools Used</b>				
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1	The Network Controller ‘checks’ the connectivity of the resources through the Resource Controller.	The Resource Controller informs the Network Controller for malfunctions in the network connectivity of the resources.		
2	The Network Controller receives the incoming messages from the Resource Controller.	The appropriate network interface is selected.		



6.1.3.3 Resource Controller (plus Navigation Service sub-component)

Table 69: Verification test of Connection and of Accuracy validation of the given Instructions

Test ID: <b>RC01</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (middle tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Connection Test and Validation of the Accuracy of the Given Instructions</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>The proxy should be connected to the testbed</li> <li>Requires the UxV to be ready to operating (e.g. en route).</li> <li>Requires the UxV to be reachable by any communication mean.</li> </ul>		
<b>Related Requirements</b>		PT-LAU-S-001, TB-PRO-001, PT-EXP-C-001, TB-MAN-001, TB-MAN-004, TB-MAN-002, TB-MAN-003, TB-MAN-005		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Receive instructions from the Experiment Controller	Instructions received		
2	Validate the Obstacle Avoidance Mechanism using known simulated scenarios	Validation Status available		
3	Validation of the Collision Avoidance Mechanism using known simulated scenarios	Validation Status available		
4	Send basic instructions to the UxVs	The UxV follows the instruction correctly, in order and timely, according to the specified parameters.		
5	Transmit the results back to the Experiment Controller			





6.1.3.4 UxV Proximity component

**Table 70: Verification test of Proximity component Backup communication**

<b>Test ID: UxP01</b>		<b>Conducted by:</b>	<b>Date:</b>	<b>Test Category: Verification Tests (UxV tier)</b>
<b>Hardware Configuration</b>		UxV with Proximity component		
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Backup communication</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>UxV are equipped with the Proximity component</li> </ul>		
<b>Related Requirements</b>		PT-GEN-001, PT-P-001, PT-P-003, PT-A-001, PT-A-003, PT-A-004, PT-A-005, PT-A-006, PT-A-007, ,PT-A-009, ,PT-A-014, PT-A-016, PT-B-001, PT-L-002, PT-E-002, PT-E-003, TB-G-004, TB-G-006, TB-I-001, TB-G-013, TB-D-001		
<b>Tools Used</b>				
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1	The UxVs are booked, the experiment is programmed and started.			
2	The UxVs lose the connection with the primary RAWFIE communication system	The Proximity communication system takes over		
3	The UxVs act autonomously, following the loaded mission instructions, logging all motion parameters, exchanging information across the swarm	The UxV use the Proximity communication system.		
4	The UxVs come back and the logged information is analysed	The communication statistics exhibits low packet error rate and low latency		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 71: Verification test of UxV retrieval using the communication system of the Proximity component**

Test ID: <b>UxP02</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (UxV tier)</b>
<b>Hardware Configuration</b>		UxV with Proximity component		
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>UxV retrieval</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>UxV are equipped with the Proximity component</li> </ul>		
<b>Related Requirements</b>		PT-GEN-001, PT-P-001, PT-P-003, PT-A-001, PT-A-003, PT-A-004, PT-A-005, PT-A-006, PT-A-007, ,PT-A-009, ,PT-A-014, PT-A-016, PT-B-001, PT-L-002, PT-E-002, PT-E-003, TB-G-004, TB-G-006, TB-I-001, TB-G-013, TB-D-001		
<b>Tools Used</b>				
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1	The UxVs are booked, the experiment is programmed and started.			
2	The UxVs perform their mission and one of them exhausts its main power source			
3	The other UxVs uses the Proximity component communication systems to communicate and locate the stopped UxV	The connection is established with the stopped UxV and the collected information allows for locating it		
4	The other UxVs transmit the location and status of the stopped UxV to the RAWFIE resource manager			

**Table 72: Verification test of Swarm motion using the Proximity component**

Test ID: <b>UxP03</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (UxV tier)</b>
<b>Hardware Configuration</b>		UxV with Proximity component		
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Swarm motion</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>UxV are equipped with the Proximity component.</li> <li>Acceptable margin for the relative location of UxV is defined depending on the type of UxV and the scenario dynamics.</li> </ul>		
<b>Related Requirements</b>		PT-GEN-001, PT-P-001, PT-P-003, PT-A-001, PT-A-003, PT-A-004, PT-A-005, PT-A-006, PT-A-007, ,PT-A-009, ,PT-A-014, PT-A-016, PT-B-001, PT-L-002, PT-E-002, PT-E-003, TB-G-004, TB-G-006, TB-I-001, TB-G-013, TB-D-001		
<b>Tools Used</b>				
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1	The UxVs are booked, the experiment is programmed and started.			
2	The UxVs perform their mission moving in a coordinated fashion			
3	The UxVs log all position			
4	The UxVs come back and the logged information is analysed	The UxV relative locations were within the acceptable margin		



6.1.3.5 Testbed Manager

Tests related to verifying Testbed Manager correct behaviour and adherence to requirements defined in D3.2 are provided in this section. The exact requirements addressed by the tests are provided in the Related Requirements field of the testing card.

Testbed Manager requirements not addressed by the tests are specified below

- TB-MAN-002,
- TB-MAN-006,
- TB-MAN-009

Test ID: <b>TM01</b>		Conducted by: <b>HAI</b>	Date:	Test Category: <b>Verification Tests (Testbed tier)</b>
<b>Hardware Configuration Details</b>				
<b>Software Configuration Details</b>				
<b>Test Name:</b>		<i>Testbed Manager Experiment Handling</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>• Requires middle tier to be accessible (Experiment Controller Service)</li> <li>• Requires local PostgreSQL Server accessible</li> </ul>		
<b>Related Requirements</b>		TB-MAN-004 TB-MAN-001 TB-MAN-005 TB-MAN-007		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Start Testbed Manager	Testbed manager successfully initialized Successful connection to the local (testbed site) database server		
2	Testbed Manager receives an ExperimentStart message from Message Bus	A new experiment is registered in the local database. Testbed Manager rejects experiments not intended for this testbed		
3	Testbed Manager receives an ExperimentStop message from Message Bus	The experiment is registered as successful in the experiments history log in the local database		
4	Testbed Manager receives an ExperimentCancel message from Message Bus	The experiment is registered as failed / partially completed in the experiments history log in the local database		
5	User selects to see the experiments executed in the testbed	Information about the experiments executed in the testbed is retrieved from the local database (experiments log) and shown in the relevant window		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

Test ID: <b>TM02</b>		Conducted by: <b>HAI</b>	Date:	Test Category: <b>Verification Tests</b> (Testbed tier)
<b>Hardware Configuration Details</b>				
<b>Software Configuration Details</b>				
<b>Test Name:</b>		<i>Manage the experiments without middle-tier connection</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Requires local PostgreSQL Server accessible</li> </ul>		
<b>Related Requirements</b>		TB-MAN-008		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	User starts Testbed Manager application in testbed site	Testbed manager successfully initialized Successful connection to the local (testbed site) database server		
2	Connection with middle-tier is lost (observed by absence of ECStatus messages received from Experiment controller in message bus)			
3	Testbed manager informs Resource Controller and initiates local storage mode	Resource controller enters in "emergency" mode Resource controller stores all sensor data from current active UxVs missions in the local database		
4	Connection with middle-tier is restored	Resource controller returns to normal mode and all sensor data are directed to RAWFIE master database		
5	Testbed manager sends all locally stored sensor data in the master database	Master database is updated with the missing data during middle-tier connection loss		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

Test ID: <b>TM03</b>		Conducted by: <b>HAI</b>	Date: <b>Feb 2016</b>	Test Category: <b>Verification Tests (Testbed tier)</b>
<b>Hardware Configuration Details</b>				
<b>Software Configuration Details</b>				
<b>Test Name:</b>		<i>Check Testbed health status</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>• Requires middle tier to be accessible (System Monitoring Service)</li> <li>• Initial Testbed Manager configuration: <ul style="list-style-type: none"> <li>○ CPU usage WARNING &gt; 50%, CRITICAL &gt;90%</li> <li>○ Memory usage WARNING &gt; 50%, CRITICAL &gt;90%</li> <li>○ Disk usage WARNING &gt; 50%, CRITICAL &gt;90%</li> <li>○ Frequency of sending messages 30 sec</li> </ul> </li> </ul>		
<b>Related Requirements</b>		TB-MAN-003		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Testbed Manager started	<ol style="list-style-type: none"> <li>1. Testbed manager successfully initialized</li> <li>2. Testbed Manager checks periodically CPU load, memory and disk usage</li> </ol>		
2	Testbed manager processing (status assessment)	<ol style="list-style-type: none"> <li>3. A TestbedHealthStatus message is created containing an overall assessment (OK, WARNING, CRITICAL) for the usage metrics monitored</li> <li>4. The message is sent to the Message bus</li> </ol>		
3	Check System monitoring Service UI display at Middle Tier	Display of Testbed Manager status. Initial status OK		
4	Artificially increase CPU or Memory usage	Status message sent to the message bus		i.e. by opening or running additional resource intensive applications in the machine where Testbed Manager is installed
5	Recheck System monitoring Service UI display at Middle Tier	Display of Testbed Manager status. Status changes to <b>WARNING</b> or <b>CRITICAL</b>		
6	Decrease CPU or Memory usage and recheck System monitoring Service UI display at Middle Tier	Display of Testbed Manager status. Status changes back to <b>OK</b>		Close extra running applications



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

Test ID: <b>TM04</b>		Conducted by: <b>HAI</b>	Date:	Test Category: <b>Verification Tests (Testbed tier)</b>
<b>Hardware Configuration Details</b>				
<b>Software Configuration Details</b>				
<b>Test Name:</b>		<i>Check the status of all services running at testbed level</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>Requires middle tier to be accessible (Experiment Controller Service)</li> <li>Requires local PostgreSQL Server accessible</li> </ul>		
<b>Related Requirements</b>		TB-MAN-003 TB-MAN-007		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	User starts Testbed Manager application in testbed site	Testbed manager successfully initialized Successful connection to the local (testbed site) database server		
2	Testbed manager receives periodical status messages from Resource Controller, Network Manager and Monitoring Manager in the Message Bus			
3	User is able to see the availability of the components that run at testbed level by selecting the appropriate action from the menu	Show current status of components running at testbed level		

**Table 73: Verification test of Testbed health status**



6.1.3.6 UxV Node

Table 74: Verification test of UxV Return to base

Test ID: UxV01		Conducted by:	Date:	Test Category: <b>Verification Tests (Testbed tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Return to base</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>- Requires the RAWFIE system to be operational (e.g. Resource controller reachable)</li> <li>- Requires the mission to be defined and running.</li> <li>- Requires the UxV to be ready to operating (e.g. en route).</li> <li>- Requires the UxV to be reachable by any communication mean.</li> </ul>		
<b>Related Requirements</b>		PT-EXA-T-008, PT-NAV-T-001, PT-NAV-T-002, PT-VIS-T-001, TB-REC-001, TB-REC-004, UXV-NET-009, UXV-SEN-003, UXV-SEN-005, UXV-PRC-001, UXV-MGT-002		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Establish the communication with the UxV	Communication established		
2	Establish a secure control session	Secured control session established		
3	Send the return to base command	Return to base command received		
4	If the UxV is not autonomous, instruct it with the necessary waypoint or guidance information, possibly until the end of the test	Further optional instructions for returning home received, Confirmation of the UxV at home		
5	Close the secure control session.	The UxV is home after a safe return. Connection closed		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 75: Verification test of the ability of the UxV to follow a route**

Test ID: UxV02		Conducted by:	Date:	Test Category: <b>Verification Tests (testbed tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Follow a route</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>- Requires the RAWFIE system to be operational (e.g. Resource controller reachable)</li> <li>- Requires the mission to be defined and running.</li> <li>- Requires the UxV to be ready to operating (e.g. en route).</li> <li>- Requires the UxV to be reachable by any communication mean.</li> </ul>		
<b>Related Requirements</b>		PT-EXA-T-008, PT-NAV-T-001, PT-NAV-T-002, PT-VIS-T-001, TB-REC-001, TB-REC-004, UXV-NET-009, UXV-SEN-003, UXV-SEN-005, UXV-PRC-001		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Step
1	Resource controller computes mission and send waypoint	Robot proceeds to the specified point,		
2	Robot continuously sends actual location	RC receives position and check if WP have been reached		
3	RC sends next point	Robot receives and proceed to next point		





## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 76: Verification test of Acquire sensor samples**

Test ID: UxV03		Conducted by:	Date:	Test Category: <b>Verification Tests (Testbed tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Acquire sensor samples</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>- Requires the RAWFIE system to be operational</li> <li>- Requires the mission to be defined and running.</li> <li>- Requires the UxV to be ready to operating (e.g. en route).</li> <li>- Requires the UxV to be reachable by any communication mean.</li> </ul>		
<b>Related Requirements</b>		PT-NF-001, UXV-SEN-005, UXV-STO-001, UXV-STO-002, UXV-NET-006, UXV-NET-007, TB-MAN-004, UXV-STO-001, UXV-STO-002, UXV-STO-003, UXV-STO-004, UXV-SEN-001, UXV-SEN-002, UXV-SEN-003, UXV-SEN-005		
<b>Tools Used</b>				
<b>Step</b>				
Step	Action	Expected Result	Status	Remarks
1	Establish the communication with the UxV	Communication established		
2	Establish a secure control session (if not done already)	Secured control session established		
3	Send the acquisition commands	Commands received and executed		
4	Store sensor samples and, if possible, transmit them via the data communication system	Samples stored and, if possible, transmitted		
5	If opened specifically for the matter of the test, close the secure control session.	Sensor samples have acquired correctly and are stored in the UxV memory or in the experiment database. Connection closed		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 77: Verification test of Fidelity to commands**

Test ID: UxV04		Conducted by:	Date:	Test Category: <b>Verification Tests (Testbed tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Fidelity to commands</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>- Requires the RAWFIE system to be operational</li> <li>- Requires the mission to be defined and running.</li> <li>- Requires the UxV to be ready to operating (e.g. en route).</li> <li>- Requires the UxV to be reachable by any communication mean.</li> </ul>		
<b>Related Requirements</b>		UXV-NET-006, UXV-NET-007, PT-NF-001, TB-MOM-003, TB-MAN-004, UXV-STO-001, UXV-STO-002, UXV-STO-003, UXV-STO-004		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Establish the communication with the UxV	Communication established		
2	Establish a secure control session (if not done already)	Secured control session established		
3	Send repeatedly pre-defined sets of commands, covering the full range of possible UxV actions,	Commands received and executed		
4	Check the conformance of the undertaken actions and corrections (if necessary) to the commands,	Undertaken actions in conformance to the commands		
5	Record all fine grained status of the UxV over the duration of the test, to be able to reconstruct the behavior of the UxV,	Status recorded		
6	If opened specifically for the matter of the test, close the secure control session.	Sensor samples have acquired correctly and are stored in the UxV memory or in the experiment database. Connection closed		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 78: Verification test of Continuous communication**

Test ID: UxV05		Conducted by:	Date:	Test Category: <b>Verification Tests (Testbed tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Continuous communication</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>- Requires the RAWFIE system to be operational</li> <li>- Requires the mission to be defined and running.</li> <li>- Requires the UxV to be ready to operating.</li> <li>- Requires the UxV to be reachable by any communication mean.</li> </ul>		
<b>Related Requirements</b>		UXV-NET-006, UXV-NET-007, TB-MOM-003, UXV-STO-004		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Establish the communication with the UxV	Communication established		
2	Exchange a predefined set of commands and data.	Commands and data correctly exchanged		
3	Close the communication session.	Communication closed		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 79: Verification test of Secure communication**

Test ID: UxV06		Conducted by:	Date:	Test Category: <b>Verification Tests (Testbed tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Secure communication</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>- Requires the RAWFIE system to be operational</li> <li>- Requires the UxV to be ready to operating.</li> <li>- Requires the UxV to be reachable by any communication mean.</li> </ul>		
<b>Related Requirements</b>		UXV-NET-006, UXV-NET-007, PT-NF-001, TB-MOM-003, UXV-STO-004		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Establish the communication with the UxV	Communication established		
2	Establish a secure control session (if not done already)	Secured control session established		
3	Check communication parameters	Communication parameters and status are correct and matching		
4	Exchange a pre-defined set of commands and data,	Commands and data correctly exchanged		
5	Close the secure control session.	Connection closed		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 80: Verification test of Real-time communication**

Test ID: <b>UxV07</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (Testbed tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Real-time communication</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>- Requires the RAWFIE system to be operational</li> <li>- Requires the mission to be defined and running.</li> <li>- Requires the UxV to be ready to operating (e.g. en route).</li> <li>- Requires the UxV to be reachable by any communication mean.</li> </ul>		
<b>Related Requirements</b>		UXV-NET-006, UXV-NET-007, PT-NF-001, TB-MOM-003, UXV-STO-004		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Establish the communication with the UxV	Communication established		
2	Establish a secure control session (if not done already)	Secured control session established		
3	Send safe commands and measure the temporal characteristics of the communication (e.g. response time, synchronisation of reception across a swarm of UxV (coordinated group of UxV), etc.).	Real-time constraints applicable to the exchanged commands are met or mismatches are detected		
4	Close the secure control session.	Connection closed		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 81: Verification test of Resume communication and data transfer**

Test ID: <b>UxV08</b>		Conducted by:	Date:	Test Category: <b>Verification Tests</b> (Testbed tier)
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Resume communication and data transfer</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>• Requires the RAWFIE system to be operational</li> <li>• Requires the mission to be defined and running.</li> <li>• Requires the UxV to be ready to operating.</li> <li>• Requires the UxV to be reachable (at least sporadically) by any communication mean.</li> </ul>		
<b>Related Requirements</b>		UXV-NET-006, UXV-NET-007, TB-MOM-003, TB-MAN-004, UXV-STO-001, UXV-STO-002, UXV-STO-003, UXV-STO-004		
<b>Tools Used</b>				
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1	Establish the communication with the UxV	Communication established		
2	Start a transaction.	Transaction started		
3	Interrupt the communication at the low-level (e.g. disconnect the antenna)	Communication is interrupted, the transaction is not complete.		
4	Re-establish the communication low level means	The transaction resumes and completes		
5	Close the communication session.	Connection closed		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 82: Verification test of UxV Device Management**

Test ID: <b>UxV9</b>	Conducted by:	Date:	Test Category: <b>Verification Tests</b> (Testbed tier)	
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>	<i>UxV Device Management</i>			
<b>Preconditions</b>	<ul style="list-style-type: none"> <li>• Requires the RAWFIE system to be operational</li> <li>• Requires the mission to be defined and running.</li> <li>• Requires the UxV to be ready to operating (e.g. en route).</li> <li>• Requires the UxV to be reachable by any communication mean.</li> </ul>			
<b>Related Requirements</b>	UXV-NET-006, UXV-NET-007, PT-NF-001, TB-MOM-003, TB-MAN-004, UXV-STO-001, UXV-STO-002,UXV-STO-003, UXV-STO-004			
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Establish the communication with the UxV	Communication established		
2	Establish a secure control session (if not done already)	Secured control session established		
3	Send device management commands	Command received and applied		
4	Check and log the status of the device	Device has responded to the commands according to the specification		
5	Close the secure control session.	The UxV is home after a safe return. Connection closed		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 83: Verification test of the UxV connection**

Test ID: <b>UxV10</b>		Conducted by:	Date:	Test Category: <b>Verification Tests (testbed tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<b>UxV Connection Test</b>		
<b>Preconditions</b>		UxV-Node launched, Message bus working		
<b>Related Requirement</b>		UXV-NET-006, UXV-NET-007, TB-MOM-003, UXV-STO-004		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Kafka Subscriber is called from another machine	Topic is shown with UxV information being published		
2	Kafka Publisher is called with a valid waypoint	Robot proceeds to the specified point		





## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 84: Verification test of Sensor Data Acquisition 1**

Test ID: UxV11		Conducted by:	Date:	Test Category: <b>Verification Tests (Testbed tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Sensor Data Acquisition 1</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>- UxV is in operation state and the parent UxV node has been launched</li> <li>- Network Communication is also fully functional</li> </ul>		
<b>Related Requirements</b>		UXV-NET-006, UXV-NET-007, PT-NF-001, TB-MOM-003, TB-MAN-004, UXV-STO-001, UXV-STO-002,UXV-STO-003, UXV-STO-004		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Establish the communication with the UxV	Communication established		
2	Establish a secure control session (if not done already)	Secured control session established		
3	Acquire sensor data	Data acquired (every sensor works as specified)		
4	Send acquired data	Data received		
5	Close the secure control session.	The UxV is home after a safe return. Connection closed		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 85: Verification test of Sensor Data Acquisition 2**

Test ID: UxV12		Conducted by:	Date:	Test Category: <b>Verification Tests (Testbed tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Sensor Data Acquisition 2</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>- UxV is in operation state and the parent UxV node has been launched</li> <li>- Network Communication is also fully functional</li> </ul>		
<b>Related Requirements</b>		UXV-NET-006, UXV-NET-007, PT-NF-001, TB-MOM-003, TB-MAN-004, UXV-STO-001, UXV-STO-002,UXV-STO-003, UXV-STO-004		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Establish the communication with the UxV	Communication established		
2	Establish a secure control session (if not done already)	Secured control session established		
3	Instruct the robot to move to a known location	Robot at the specific location		
4	Acquire current location data	Location data acquired (location sensor works as specified)		
5	Send acquired location data	Data received		
6	Close the secure control session.	The UxV is home after a safe return. Connection closed		



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

**Table 86: Verification test of Data Storage**

Test ID: UxV13		Conducted by:	Date:	Test Category: <b>Verification Tests (Testbed tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Data Storage</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>- UxV is in operation state and the parent UxV node has been launched.</li> <li>- Sensor node is functional</li> </ul>		
<b>Related Requirements</b>		UXV-NET-006, UXV-NET-007, TB-MAN-004, UXV-STO-001, UXV-STO-002, UXV-STO-003, UXV-STO-004, TB-MAN-004, UXV-STO-001, UXV-STO-002, UXV-STO-003, UXV-STO-004,		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Establish the communication with the UxV	Communication established		
2	Establish a secure control session (if not done already)	Secured control session established		
3	A request for storing certain data is done	Command received and data is stored locally		
4	After a mission given, data storage in the system is checked.	Data was correctly stored and kept.		
5	Close the secure control session.	The UxV is home after a safe return. Connection closed		



**Table 87: Verification test of Waypoints Processed**

Test ID: UxV14		Conducted by:	Date:	Test Category: <b>Verification Tests (Testbed tier)</b>
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>		<i>Waypoints Processed</i>		
<b>Preconditions</b>		<ul style="list-style-type: none"> <li>- UxV is in operation state and the UxV parent node has been launched.</li> <li>- Sensor node is functional, network communication is functional</li> </ul>		
<b>Related Requirements</b>		UXV-NET-006, UXV-NET-007, TB-MAN-004, UXV-STO-001, UXV-STO-002, UXV-STO-003, UXV-STO-004,		
<b>Tools Used</b>				
Step	Action	Expected Result	Status	Remarks
1	Establish the communication with the UxV	Communication established		
2	Establish a secure control session (if not done already)	Secured control session established		
3	Waypoints are sent to the UxV	UxV receives and processes the waypoints		
4	The calculated route is applied to the UxV	The actual trajectory matches the route calculated by the navigation.		
5	Iterate step 4 until assessment is complete	UxV stops, informs and recalculate its route to next waypoint if an unexpected obstacle is found.		
6	Close the secure control session.	The UxV is home after a safe return. Connection closed		

## 6.2 Integrated system testing

As well as testing each individual component, the system will also be tested as a whole unit to validate its overall behaviour. Testing will be covered in the following areas:

The integrated testing procedure will be detailed during the first development iteration. The testing procedure will be based on the successful chain of verification scenarios described in Section 2 of this document.

Such scenarios will correspond to sequences and combinations of the components tests.

## 7 Validation scenarios

This chapter describes the validation scenarios. Some have been defined by the selected users of the RAWFIE system. Other simpler and more dedicated scenarios can focus on the evaluation of specific characteristics or behaviours of the RAWFIE components, testbeds, federation, etc. They are defined on the basis of requirements described in D3.1 and D3.2. Other scenarios may be defined on the basis of user defined use cases.



## 7.1 User defined scenarios

In the first version of requirements' deliverable (D3.1) a set of user scenarios were defined with the purpose to serve as a starting point for identifying the needs and assisting the elicitation of high level system wide requirements, for the potential experiments that should be supported by the platform. D3.2 added two further scenarios. From the evaluation of the 1<sup>st</sup> Open Call also several new scenarios were derived.

These user defined scenarios can be considered as a starting point for the definition of appropriate activities and steps that can be used for the overall RAWFIE platform validation. Despite their differences in nature and purpose, when considering them from the RAWFIE platform perspective, a set of common general steps can be identified for all of them. These general steps are summarized below:

1. The experimenter looks for a UxV testbed where the UxVs could be or are equipped with the technology T (e.g. infra-red cameras, ZigBee transmitters, radar, etc.) and the testbed provides an environment E.
2. The experimenter books resources in a testbed for the desirable timeframe.
3. The experimenter writes the experiment steps with EDL. Depending on the experimenter an algorithm A may be declared in EDL using the provided API.
4. The experiment is scheduled for execution at the given timeframe (actual resources are associated with it during this step)
5. UxV gets equipped with technology T by the support personal, if necessary.
6. The experiment is launched
7. UxVs execute the given script correctly. Declared algorithm A is carried out.
8. Measurements M are sent to message bus, evaluated by the algorithms and stored in the database.
9. Experimenter observes the experiment via the appropriate platform services (Experimenter monitoring Tool, Visualization Tool) and can intervene to the execution if need be..
10. The experiment completes.
11. The experimenter evaluates the results/measurements and possibly assesses the behavior of the applied algorithm A and technology T through the appropriate platform services (experiment log, Data Analysis Tool, etc.)

It becomes evident that the differences will be in the used technologies (T), recorded measurements (M), the testbed environment (E) and the possible algorithms (A) that need to be reflected in EDL.



The involved subsystems will almost always be:

- Resource Explorer Tool
- Testbeds Directory Service
- Booking Tool
- Booking Service
- Experiment Authoring Tool
- EDL Compiler & Validator
- Experiment Validation Service
- Launching Service
- Experiment Monitoring Tool
- Visualization Tool
- Visualization Engine
- Data Analysis Tool
- Data Analysis Engine

### **7.1.1 Monitoring of Water Canals**



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

<b>Scenario ID: UD-01</b>		Conducted by:		Date:	
<b>Title</b>		Monitoring of Water Canals			
<b>Comment</b>		UxVs that can collaborate for the purpose of environmental monitoring of water canals and gather of information that can be used for assessing quality of the water and structural integrity of canal walls <i>See also: D3.1 section 3.3.1</i>			
<b>Validated requirement</b>					
<b>Technology</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
Fixed wing UAV		inspect rapidly a large area.			
Rotary wing UAV		inspect precisely the problematic area			
USV or UUV		inspect precisely the underwater problem area			
UGV		inspect precisely bank areas			
spectral imaging sensor and areal camera		image the area via UAV			
bathymetric sensor (sound sensors)		acoustic maps of the underwater area via USV or UUV			
<b>Measurements</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
Spectral images					
Areal images					
Acoustic maps					
<b>Environment</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
Open air water channels		These channels should be able to be prepared to simulate a pollution			
<b>Algorithm</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
Image analysis		Identify problems on spectral images, areal images and acoustic maps			
Movement pattern		<ul style="list-style-type: none"> <li>- Evaluate patterns for inspecting rapidly of a large area via fixed wing UAV</li> <li>- Evaluate patterns for inspect precisely of a small area via rotary wing UAV</li> </ul>			
<b>Special script steps</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
<b>Metric</b>		<b>Success criteria</b>	<b>Status</b>	<b>Remarks</b>	
All					



**7.1.2 Border Surveillance or Perimeter protection of large areas**

Scenario ID: <b>UD-02</b>	Conducted by:	Date:	
<b>Title</b>	Border Surveillance or Perimeter protection of large areas		
<b>Comment</b>	UxVs that can collaborate for the purpose of border, infrastructure or sensitive area monitoring and gather information that can be used for assessing a potential threat and take urgent action to protect the area or borders from invention or asymmetric threats. It includes maritime border surveillance. <i>See also:</i> D3.1 section 3.3.2		
<b>Validated requirement</b>			
<b>Technology</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
UAV	observation of the area		
UGV	observation of the area		
	play the intruder		
Day/night thermal cameras	Mounted on UAVs and UGVs for intruder detection		
Radars			
CBNR sensors			
Acoustic sensors			
<b>Measurements</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
Thermal images			
Radar maps			
CBNR measurements			
Acoustic measurements			
<b>Environment</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
Open air terrestrial area			
<b>Algorithm</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
Intruder detection	Based on all the sensor values, intruders should be detected		
<b>Special script steps</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
Search for large testbed area with many UAV, UGV	Experimenter looks for a UAV and UGV testbed with an large testbed area and many available UAV and UGV via the Resource Explorer Tool		
Many UxVs simultaneously needed.	The Booking Tool supports the Experimenter to find a date where enough UxVs are available		
EDL script	<ol style="list-style-type: none"> <li>1. UGVs should go to their starting position</li> <li>2. UAVs and UGVs start the detection of intruders at the perimeter</li> <li>3. UGVs start “intruding”</li> <li>4. UAVs and UGVs execute the detection program</li> </ol>		
<b>Metric</b>	<b>Success criteria</b>	<b>Status</b>	<b>Remarks</b>
All			





7.1.3 On demand deployable Internet facilities

Scenario ID: <b>UD-03</b>	Conducted by:	Date:	
<b>Title</b>	On demand deployable Internet facilities		
<b>Comment</b>	This validation scenario tackles the rapidly expanding domain of on-demand deployable Internet facilities through UxVs. The experimenter has a program/algorithm for UAVs that provide internet access over the air. Includes Deployment of mesh networks in emergency contexts for purposes of providing communications and connectivity. Provide Internet access in remote areas. <i>See also: D3.1 section 3.3.3</i>		
<b>Validated requirement</b>			
<b>Technology</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
UAV	Setup network hubs		
UGV	Setup network hubs		
	Play a network client causing traffic		
Wi-Fi	Access point for clients		
TBD wireless communication	Communication between network hubs		
<b>Measurements</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
Wi-Fi signal strength	Determine the network coverage		
Network throughput			
Energy consumption			
<b>Environment</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
<b>Algorithm</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
Scheduling algorithms	tasking of UxVs: like <ul style="list-style-type: none"> <li>- channel bonding by several UxVs with uplink</li> <li>- spreading in the area</li> <li>- energy saving (e.g. landing of UAVs, switch-off underexploited UxVs)</li> </ul>		
Network routing protocols	Routing based on usage, availability and priority.		
<b>Special script steps</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
Search for large testbed area with many UAV, UGV	Experimenter looks for a UAV and UGV testbed with many available UAV and UGV via the Resource Explorer Tool		
<b>Metric</b>	<b>Success criteria</b>	<b>Status</b>	<b>Remarks</b>
All			



**7.1.4 Exploration & Assessment of Network Technologies Robustness**

Scenario ID: <b>UD-04</b>		Conducted by:		Date:	
<b>Title</b>		Exploration & Assessment of Network Technologies Robustness			
<b>Comment</b>		This validation scenario aims to assess the networking performance and robustness with respect to certain parameters and factors (i.e., communications range, throughput, error distribution) and support a subsequent exploration analysis for identifying the best deployment per case basis. <i>See also:</i> D3.1 section 3.3.4			
<b>Validated requirement</b>					
<b>Technology</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
UGV, UAV		Setup network hubs			
Different wireless communication		wireless communication technology to be tested			
<b>Measurements</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
Distributions of errors					
Signal-to-noise ratio (SNR)					
Throughput					
Disconnections					
<b>Environment</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
Area with different obstacles		Assess the influences of obstacles			
<b>Algorithm</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
maximization of network coverage		Maximization of network coverage based on - Obstacles - Number of UxVs - Computation needs and battery lifetime			
Switching of wireless communication technology		Switch to alternative network interfaces whenever need arises.			
<b>Special script steps</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
<b>Metric</b>		<b>Success criteria</b>	<b>Status</b>	<b>Remarks</b>	
All					



**7.1.5 Efficient Coordination for phenomena or mission**

Scenario ID: <b>UD-05</b>		Conducted by:		Date:	
<b>Title</b>		Efficient Coordination for phenomena or mission			
<b>Comment</b>		This validation scenario aims to assess the performance of the coordination a set of UxVs to observe specific phenomena or to conclude specific missions. <i>See also:</i> D3.1 section 3.3.5			
<b>Validated requirement</b>					
<b>Technology</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
UAV, UGV					
Some sensor that detects a phenomena		Sensor that detects e.g. a fire that should be investigated further			
<b>Measurements</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
Resources consumption		Resources (UxV, battery, etc.) needed until mission completed			
Time		e.g. time taken from initial detection to attain max coverage (Reactivity)			
Spatial coverage		Area covered at different time steps during the experiment			
<b>Environment</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
<b>Algorithm</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
Swarm coordination		Self-organization and reorganization to more accurately capture the spatiotemporal development of the phenomenon			
Collaborative mission		<ul style="list-style-type: none"> <li>- Negotiate a scanning plan within the swarm and set out for the scanning area</li> <li>- Following completion of the sensing task the UxV return to network reachability and communicate cached measurements to ground control (onboard data caching)</li> </ul>			
<b>Special script steps</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
<b>Metric</b>		<b>Success criteria</b>	<b>Status</b>	<b>Remarks</b>	
All					



**7.1.6 Over the Air (OTA) UxV Re-programming**

Scenario ID: <b>UD-06</b>		Conducted by:		Date:	
<b>Title</b>		Over the Air (OTA) UxV Re-programming			
<b>Comment</b>		The aim of this scenario is to use the OTA re-programming procedure that will be triggered by nodes, after a failure occurs or when extensions on board applications are required. <i>See also: D3.1 section 3.3.6</i>			
<b>Validated requirement</b>					
<b>Technology</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
UxV		Any UxV type could be used			
Onboard application supervision		Module which has complete control over the OTA staging area and can invoke specific methods (callback) in the involved applications (e.g., termination, startup). Used to update algorithms/programs on the UxV			
<b>Measurements</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
Changing of measurements		After updating the algorithms/programs, the UxV should send other sensor values.			
<b>Environment</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
Different sensor processing algorithms		Different sensor processing algorithms that can be updated on the UxV			
<b>Algorithm</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
Different sensor processing algorithms		Different sensor processing algorithms that can be updated on the UxV			
<b>Special script steps</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
<b>Metric</b>		<b>Success criteria</b>	<b>Status</b>	<b>Remarks</b>	
All					

**7.1.7 Additional scenarios from Open calls**

In addition, the first Open call led to proposals involving new usages, use case and user scenarios that have been used for completing the existing user defined scenarios or for defining new scenarios.



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

<b>Scenario ID: UD-07</b>		<b>Conducted by:</b>		<b>Date:</b>	
<b>Title</b>		2. Navigation, autopilot, communication and obstacle avoidance system			
<b>Comment</b>		The aim of this scenario is to exercise the navigation, autopilot, communication and obstacle avoidance systems			
<b>Validated requirement</b>		Open call 1			
<b>Technology</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
UxV		Any UxV type could be used			
Sensors for obstacle detection		Sensors like cameras, laser scanners, radars, etc.			
GPS etc		Positions of the UxV			
<b>Measurements</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
Images					
Position data					
Distance maps					
<b>Environment</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
<b>Algorithm</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
Obstacle avoidance		Stress the UxV Algorithm to avoid obstacle using its embedded sensing means.			
<b>Special script steps</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
<b>Metric</b>		<b>Success criteria</b>	<b>Status</b>	<b>Remarks</b>	
All					



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

<b>Scenario ID: UD-08</b>	<b>Conducted by:</b>	<b>Date:</b>	
<b>Title</b>	8. Flight monitoring on the UAV tested into his airspaces, beyond peer to peer and standard GNSS quality		
<b>Comment</b>	embed several communication protocols and frequencies into the tracking system plus a selective accurate GNSS components		
<b>Validated requirement</b>	Open call 1		
<b>Technology</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
UAV			
Sensors for positioning	Possibility to compare different technologies (e.g. accuracy of the positioning)		
<b>Measurements</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
Position raw data	Raw data for positioning determination		
<b>Environment</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
Various	Various environment should be tested to determine their effect on the different positioning technologies		
<b>Algorithm</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
Positioning determination	Compute the position of the UxV out of the sensor values		
<b>Special script steps</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
<b>Metric</b>	<b>Success criteria</b>	<b>Status</b>	<b>Remarks</b>
All			



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

<b>Scenario ID: UD-09</b>	Conducted by:	Date:	
<b>Title</b>	9. The geofencing service		
<b>Comment</b>	The geofencing service will be improved, providing a better security for the operations as well as a good quality of the data for the flights logs analysis by the experimenters. The SES component will be evaluated related to the geofencing service performance.		
<b>Validated requirement</b>	Open call 1		
<b>Technology</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
Long distance drones	Experimenters using CESA-Drones sites will have the possibility to perform various tests such as “long distance” scenario (up to 50 km in a segregated area)		
Geofencing capability	Experimenters will access to high end tools such as performant and accurate geofencing		
Safety security of the platform (ethics)			
<b>Measurements</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
Positions	Position of the UAVs		
<b>Environment</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
“Long distances”	Long distance flights should be possible inside the testbed		
<b>Algorithm</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
<b>Special script steps</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
<b>Metric</b>	<b>Success criteria</b>	<b>Status</b>	<b>Remarks</b>
All			



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

<b>Scenario ID: UD-10</b>	<b>Conducted by:</b>	<b>Date:</b>	
<b>Title</b>	15. Large scale imaging and 3D visualization or aerial photography		
<b>Comment</b>	Includes real-time visual Intelligence		
<b>Validated requirement</b>	Open call 1		
<b>Technology</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
Position, GNSS	Long-term flights, high data throughput, precise GNSS, collaboration between UxVs for efficiency		
UAV	Long-term flights		
Cameras	For aerial photography		
<b>Measurements</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
aerial photography			
UAV positions			
<b>Environment</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
Outdoor			
<b>Algorithm</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
3D visualization			
Large scale imaging			
UAV collaboration	collaboration between UxVs for efficiency		
<b>Special script steps</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
<b>Metric</b>	<b>Success criteria</b>	<b>Status</b>	<b>Remarks</b>
All			





## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

<b>Scenario ID: UD-11</b>	Conducted by:	Date:	
<b>Title</b>	16. 17. 18. 19 Safety inspection for fire-fighters, police, anti-terrorist and industrial environments		
<b>Comment</b>	Special police or anti-terrorist indoor interventions, with an aerial overview for identification and localization and threat localization purposes Indoor safety inspection with UGV or UAV Industrial inspection with a USV UAV or UGV robots USV Indoor Fire search and rescue operations: nodes, relays, communications using UxVs,		
<b>Validated requirement</b>	Open call 1		
<b>Technology</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
UxV	Different UxV types possible, based on the concrete experiment.		
Detection sensors	Detection of threat		
Localisation sensors	Localisation of UxV		
Identification sensor			
Communication transceivers	Thick walls and their inherent communication issues		
<b>Measurements</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
UxV positions			
Sensor values to identify threats	Depends on the used sensor and threads to be identified.		
<b>Environment</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
<b>Algorithm</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
Identification algorithms	Identification of threats out of sensor values		
<b>Special script steps</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
<b>Metric</b>	<b>Success criteria</b>	<b>Status</b>	<b>Remarks</b>
All			



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

<b>Scenario ID: UD-12</b>		<b>Conducted by:</b>		<b>Date:</b>	
<b>Title</b>		Monitoring and reporting on a variety of parameters, ranging from weather conditions and environmental measurements to detection of movement/intrusions and more.			
<b>Comment</b>					
<b>Validated requirement</b>		Open call 1			
<b>Technology</b>					
<b>Technology</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
Detection sensors		Detection of movement			
Localisation sensors		Localisation of UxV			
Identification sensor					
<b>Measurements</b>					
<b>Measurements</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
Sensor values to identify movement source		Depends on the used sensor and threads to be identified.			
<b>Environment</b>					
<b>Environment</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
<b>Algorithm</b>					
<b>Algorithm</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
Identification algorithms		Identification of movement source out of sensor values			
<b>Special script steps</b>					
<b>Special script steps</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
<b>Metric</b>					
<b>Metric</b>		<b>Success criteria</b>	<b>Status</b>	<b>Remarks</b>	
All					

The above scenario corresponds aims at monitoring and reporting on a variety of parameters, ranging from weather conditions and environmental measurements to detection of movement/intrusions and more.



7.1.8 Efficient coordination of multiple UxVs

Scenario ID: <b>UD-13</b>		Conducted by:		Date:	
<b>Title</b>		Efficient coordination of multiple UxVs			
<b>Comment</b>		This scenario deals with the efficient coordination of multiple UxVs for the purpose of covering certain phenomena (e.g. fire spreading in an area) or executing a certain sensing mission (e.g. mapping or scanning of an unknown area).			
<b>Validated requirement</b>		PT-GEN-001, PT-P-001, PT-P-003, PT-A-001, PT-A-003, PT-A-004, PT-A-005, PT-A-006, PT-A-007, ,PT-A-009, ,PT-A-014, PT-A-016, PT-B-001, PT-L-002, PT-E-002, PT-E-003, TB-G-004, TB-G-006, TB-I-001, TB-G-013, TB-D-001			
<b>Technology</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
UxV		Any UxV type could be used (group of UxV resources)			The experimenter wants to test an algorithm for spatial coverage of an area of interest with the minimum energy consumption
<b>Measurements</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
Node lifetime		How long was a UxV online before battery was to low			
Nodes energy consumption					
Node positions		The positions that were used to cover the area			
Percentage of the covered area					
<b>Environment</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
Testbed		UxV-specific testbed (e.g. Surface, Aerial, Ground, etc.)			
<b>Algorithm</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
Swarm algorithm		Different swarm algorithms to be evaluated.			From experiment log and examine measurements: Percentage of the covered area, Nodes lifetime, Nodes energy consumption, Final positions
<b>Special script steps</b>		<b>Details</b>		<b>Status</b>	<b>Remarks</b>
EDL script		<ul style="list-style-type: none"> <li>• Defines a specific area of interest</li> <li>• Defines the algorithm for the coordination of the UxVs (like Particle Swarm Optimization algorithm)</li> </ul>			
<b>Metric</b>		<b>Success criteria</b>	<b>Status</b>	<b>Remarks</b>	



## 7.2 RAWFIE Platform Admin scenarios

### 7.2.1 Administrator manages the user rights

<b>Scenario ID: PA-01</b>		<b>Conducted by: Fraunhofer</b>		<b>Date: Feb 2016</b>	
<b>Title</b>		Administrator manages the user rights			
<b>Comments</b>					
<b>Main stakeholder</b>		RAWFIE Admin			
<b>Secondary stakeholder</b>		Experimenters			
<b>Involved Sub-systems</b>		Web Portal Users & Rights Service			
<b>Validated requirement</b>		PT-GEN-001, PT-GEN-002			
<b>Step</b>	<b>Description</b>	<b>Status</b>	<b>Remarks</b>		
1	Administrator opens the user management of the Web Portal	Not tested			
2	Administrator searches for a given user	Not tested			
3	Administrator changes the rights of the given user	Not tested			
<b>Metric</b>		<b>Success criteria</b>	<b>Status</b>	<b>Remarks</b>	

### 7.2.2 Administrators adds a new user

<b>Scenario ID: PA-02</b>		<b>Conducted by: Fraunhofer</b>		<b>Date: Feb 2016</b>	
<b>Title</b>		Administrators adds a new user			
<b>Comments</b>					
<b>Main stakeholder</b>		RAWFIE Admin			
<b>Secondary stakeholder</b>		Experimenters			
<b>Involved Sub-systems</b>		Web Portal Users & Rights Service			
<b>Validated requirement</b>		PT-GEN-001, PT-GEN-002			
<b>Step</b>	<b>Description</b>	<b>Status</b>	<b>Remarks</b>		
1	Administrator opens the user management of the Web Portal				
2	Administrator clicks on “new user”				
3	Administrator inserts the user data and submits the data				
4	Users & Rights Service save the user				
5	Information is sent to the new user via email				
<b>Metric</b>		<b>Success criteria</b>	<b>Status</b>	<b>Remarks</b>	



### 7.2.3 System monitoring and error notifications

<b>Scenario ID: PA-03</b>		<b>Conducted by: Fraunhofer</b>		<b>Date: Feb 2016</b>	
<b>Title</b>		System monitoring and error notifications			
<b>Comments</b>					
<b>Main stakeholder</b>		RAWFIE Admin			
<b>Secondary stakeholder</b>					
<b>Involved Sub-systems</b>		Web Portal System Monitoring Tool System Monitoring Service (Launching Service)			
<b>Validated requirement</b>		PT-GEN-001, PT-GEN-002			
<b>Step</b>	<b>Description</b>	<b>Status</b>	<b>Remarks</b>		
1	Launching Service crashes	n.a.			
2	System Monitoring Service checks system state and detects that Launching Service is not running		Launching Service not monitorable until now		
3	System Monitoring Service sends a notification email to the administrator		Email sending not configured		
4	Administrator opens the System Monitoring Tool				
5	Administrator checks system state				
6	Administrator restarts Launching Service via some SSH client				
7	Administrator checks system state (now Launching Service is running again)				
<b>Metric</b>		<b>Success criteria</b>	<b>Status</b>	<b>Remarks</b>	
PLATFORM / PERF / 1 / STABLE SYSTEM					
PLATFORM / PERF / 2 / ERRORS					
PLATFORM / PERF / 4 / RECOVERY TIME					
PLATFORM / USE / 7 / NOTIFICATION					
PLATFORM / USE / 10 / VISUALISATION / SIMPLICITY					
PLATFORM / USE / 12 / VISUALISATION / UTILITY					
PLATFORM / USE / 13 / GUIDANCE					
PLATFORM / USE / 14 / FILTERING					

## 7.3 Testbed operator scenarios

### 7.3.1 Schedule maintenance of resources



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

<b>Scenario ID: TO-01</b>		Conducted by:	Date:
<b>Title</b>		Schedule maintenance	
<b>Comment</b>		The Testbed operator wants, for maintenance purposes, to temporary remove some resources (UxVs) already assigned to future experiments from a testbed	
<b>Main stakeholder</b>		Testbed Operator	
<b>Secondary stakeholder</b>		Experimenters	
<b>Involved Sub-systems</b>		Web Portal Booking Tool Booking Service Testbed Directory Service Users & Rights Service	
<b>Validated requirement</b>		PT-GEN-R-002, PT-BOO-T-003, PT-BOO-T-004, PT-BOO-T-005, PT-BOO-T-006, PT-BOO-T-008, PT-BOO-T-009, PT-BOO-T-010, PT-BOO-S-001, PT-BOO-S-002, PT-BOO-S-005, PT-BOO-S-007, PT-BOO-S-011, PT-DIR-S-003, PT-DIR-S-004, PT-DIR-S-006, PT-USR-S-001, PT-USR-S-002,	
<b>Step</b>	<b>Description</b>	<b>Status</b>	<b>Remarks</b>
1	Testbed operator wants to maintain certain UxVs because a problem has occurred		
2	Via the Booking Tool he tries to find a period where the involved UxVs are free		
3	He could not find one in the near future and decides to cancel some bookings		
4	The affected experimenters are notified via email that their bookings were cancelled		
5	The involved UxVs become unavailable for the period of the planned maintenance		
6	A new experimenter trying to make a Booking to the specified testbed should not be able to select the unavailable UxVs		
7			
<b>Metric</b>		<b>Success criteria</b>	<b>Status</b>
PLATFORM / USE / 7 / NOTIFICATION			
PLATFORM / USE / 8 / ROLES			
PLATFORM / USE / 10 / VISUALISATION / SIMPLICITY			
PLATFORM / USE / 12 / VISUALISATION / UTILITY			
PLATFORM / USE / 13 / GUIDANCE			
PLATFORM / USE / 14 / FILTERING			
<b>TESTBED / DATA / 1 / INFORMATION</b>			

### 7.3.2 Cancel running experiment



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

<b>Scenario ID: TO-02</b>	Conducted by:	Date:	
<b>Title</b>	Cancel running experiment		
<b>Comment</b>	A testbed operator figures erroneous behavior and wants to cancel a running experiment and ensure the resources return safely to their base		
<b>Main stakeholder</b>	Testbed Operator		
<b>Secondary stakeholder</b>	Experimenters (e.g. via the Experiment Monitoring tool and Experiment Controller)		
<b>Involved Sub-systems</b>	Web Portal Experiment Monitoring Tool Launching Service Experiment Controller Navigation Service Resource Controller Visualization Tool		
<b>Validated requirement</b>	PT-EXM-T-001, PT-EXM-T-002, PT-EXM-T-003, PT-NAV-T-003, PT-LAU-S-010, PT-LAU-S-012, PT-EXP-C-001, PT-EXP-C-007, PT-EXP-C-008, PT-EXP-C-009, TB-REC-002, TB-REC-003, TB-REC-006, PT-VIS-T-001, PT-VIS-E-001,		
<b>Step</b>	<b>Description</b>	<b>Status</b>	<b>Remarks</b>
1	the Testbed Operator notices that something goes wrong		
2	he opens the Experiment Monitoring Tool and browse to the experiment		
3	he initiate the cancelation of the experiment via the Experiment Monitoring Tool		
4	the Experiment Monitoring Tool instructs the Experiment Controller (via Launching Service)		
5	the Experiment Controller issues the appropriate commands to send the UxVs back to the port		
6	the Resource Controller receives the commands and guides the UxVs back (possible activation of emergency scenario).		
7	The Testbed Operator is able to view the route of UxV on a map and confirm that it returned to base		
<b>Metric</b>	<b>Success criteria</b>	<b>Status</b>	<b>Remarks</b>
PLATFORM / USE / 7 / NOTIFICATION			
PLATFORM / USE / 8 / ROLES			
PLATFORM / USE / 10 / VISUALISATION / SIMPLICITY			
PLATFORM / USE / 12 / VISUALISATION / UTILITY			
PLATFORM / USE / 13 / GUIDANCE			
PLATFORM / USE / 14 / FILTERING			
TESTBED / DATA / 1 / INFORMATION			



### 7.3.3 Connect a new Testbed to the RAWFIE platform

<b>Scenario ID: TO-03</b>		Conducted by:	Date:
<b>Title</b>		Connect a new testbed	
<b>Comment</b>			
<b>Main stakeholder</b>		Testbed Operator	
<b>Secondary stakeholder</b>		RAWFIE Admin	
<b>Involved Sub-systems</b>		Web Portal Experiment Monitoring Tool Experiment Controller Navigation Service	
<b>Validated requirement</b>			
<b>Step</b>	<b>Description</b>	<b>Status</b>	<b>Remarks</b>
1	The Testbed Operator agrees with the RAWFIE platform Admin to connect its Testbed		
2	Testbed Operator ensures the testbed fulfil the needed requirements to be connected to the RAWFIE platform (Networking facilities, and so on)		
3	Testbed Operator updates the Master Data Repository with new Testbed information via the Resource Explorer		
4	Testbed Operator configures the Testbed components to be able to communicate with the rest of the RAWFIE platform		
<b>Metric</b>		<b>Success criteria</b>	<b>Status</b>
PLATFORM / USE / 7 / NOTIFICATION			
PLATFORM / USE / 8 / ROLES			
PLATFORM / USE / 10 / VISUALISATION / SIMPLICITY			
PLATFORM / USE / 12 / VISUALISATION / UTILITY			
PLATFORM / USE / 13 / GUIDANCE			
PLATFORM / USE / 14 / FILTERING			
TESTBED / DATA / 1 / INFORMATION			
PLATFORM / FUNC / 17 / EXTENSIBILITY			

## 7.4 UxV Manufacturers scenarios

### 7.4.1 Install new UxVs in a testbed





<b>Scenario ID: UM-01</b>		<b>Conducted by:</b>	<b>Date:</b>
<b>Title</b>		Install new UxVs in a testbed	
<b>Comment</b>			
<b>Main stakeholder</b>		UxV Manufacturers	
<b>Secondary stakeholder</b>		Testbed Operator	
<b>Involved Sub-systems</b>		Web Portal Resource Explorer	
<b>Validated requirement</b>		PT-P-003, TB-G-004	
<b>Step</b>	<b>Description</b>	<b>Status</b>	<b>Remarks</b>
1	UxV Manufacturer ask the Testbed Operator if new UxVs could be installed in the testbed		
2	Testbed Operator agrees		
3	UxV Manufacturer sends the new UxVs to the testbed site		
4	UxV Manufacturer give the information about the UxVs to the Testbed Operator		
5	Testbed Operator update the resource description for its testbed via the Resource Explorer		
6	UxV Manufacturer ensures the UxV Node is able to send / receive information to from the RAWFIE components through the foreseen software interfaces		
7	UxV Manufacturer and Testbed Operator configure the Testbed and RAWFIE platform components to control the new UxVs		
<b>Metric</b>		<b>Success criteria</b>	<b>Status</b>
PLATFORM / FUNC / 17 / EXTENSIBILITY			
PLATFORM / USE / 7 / NOTIFICATION			
PLATFORM / USE / 8 / ROLES			
PLATFORM / USE / 10 / VISUALISATION / SIMPLICITY			
PLATFORM / USE / 12 / VISUALISATION / UTILITY			
PLATFORM / USE / 13 / GUIDANCE			
PLATFORM / USE / 14 / FILTERING			

**7.4.2 Autonomous coordination of multiple UxVs**



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

<b>Scenario ID: UM-02</b>		<b>Conducted by:</b>	<b>Date:</b>
<b>Title</b>		Autonomous coordination of multiple UxVs	
<b>Comment</b>		This scenario deals with the autonomous coordination of multiple UxVs for providing the RAWFIE experiment with some robustness with respect to the loss of communication or performance issue in the connection between the UxV swarms and the RAWFIE system. This is particularly relevant for ensuring the UxV coordination when they are operating in large remote areas or over the sea.	
<b>Main stakeholder</b>		Testbed Manager, Experimenters	
<b>Secondary stakeholder</b>		UxV Manufacturers,	
<b>Involved Sub-systems</b>		Local RAWFIE entities Proximity component	
<b>Validated requirement</b>		TB-UVG-001, UXV-MGT-002, UXV-NET-002, UXV-NET-003, UXV-NET-004, UXV-NET-005, UXV-NET-006, UXV-NET-007, UXV-NET-008, UXV-NET-009, UXV-PRC-001, UXV-PRC-002, UXV-PRC-004, UXV-MGT-002, UXV-MGT-004, UXV-MGT-005, UXV-NOD-001	
<b>Step</b>	<b>Description</b>	<b>Status</b>	<b>Remarks</b>
1	The UxV manufacturer(s) deploys several UxVs that will operate in swarm in the experiment. The experiment consists in collecting and analysing the data exchanges that occurred during the experiment on the Proximity component network interface, for the sake of the coordination of the UxV motion.		
2	UxV Manufacturer sends the new UxVs to the testbed site. UxV Manufacturer gives the information about the UxVs to the Testbed Operator.		
3	Testbed Operator update the resource description for its testbed via the Resource Explorer, while the route followed by the UxV and relative UxV locations are specified in the experiment EDL script. UxV Manufacturer and Testbed Operator configure the testbed to control the new UxVs.		
4	The experiment is started and the experimental conditions, the exchanged data and the behavior of the UxV are logged with a time information.		
5	The UxV manufacturer collects the logged data and evaluates the relationship between the experimental conditions, the exchanged data and the behaviour of the UxV <ul style="list-style-type: none"> <li>• View experiment log</li> <li>• Examine measurements</li> <li>• Percentage of the covered area</li> <li>• Nodes lifetime</li> <li>• Nodes energy consumption</li> <li>• Final positions</li> </ul>		
6	The experimenter details the deviations of the UxV route and their relative trajectories from the expected behaviour.		
7			
<b>Metric</b>		<b>Success criteria</b>	<b>Status</b>
All			Communication and networking behaviour, data exchange statistics, including temporal characteristics and Swarm flying



## 7.5 Early sub-system tests and validation

Matching pilot experimentation scenarios for validation to the use cases described in D3.1 one-to-one postpones testing for validation to a very late stage of project development and requires a lot of resources. Even though RAWFIE focuses on large scale experimentation of real UxVs, it is envisaged to show some evidence that the RAWFIE platform works well in smaller scale experiments or with a reduced set of functions or components.

As a consequence of the above, at least two additional pilot experimentation scenarios have been introduced to allow for early tests and validation of sub-systems or reduced scale RAWFIE systems.

Both cases assume that all Front-end tier, middle tier and data tier components are fully functional and running. The end user can write and launch validated experiments which can be conducted using limited or no UxV resources.

In the future this section may be augmented with additional tests needed to validate the correctness of different UxVs subsystems integration to RAWFIE platform prior the phase of executing the end-user defined validation scenarios as described in the previous sections.



7.5.1 UxV Data Generator

<b>Scenario ID: EST-01</b>	<b>Conducted by:</b>	<b>Date:</b>
<b>Title</b>	UxV Data Generator	
<b>Comment</b>	An “UxV Data Generator” component is implemented in the lower layer of Testbed and feeds the system with messages identical the ones generated from the UxV resources. A suitable log file also verifies that commands/responses from the RAWFIE platform arrive in testbed tier in the expected format. The “UxV Data Generator” component simulates to an extent the behaviour of an UxV device implementing incrementally from basic to more complex features. The scope of this validation scenario is to give to the experimenter the ability to write and run experiments in the RAWFIE platform in the absence of UxV resources and validate that the steps of the experiment are executed in the order and time specified in the scripts.	
<b>Main stakeholder</b>	Experimenter	
<b>Secondary stakeholder</b>	RAWFIE Platform Administrator / Testbed Operators /UxVs Manufacturers	
<b>Involved Sub-systems</b>	Web Portal Users & Rights Service Resource Explorer Tool Testbeds Directory Service Experiment Authoring Tool EDL Compiler & Validator Experiment Validation Service Booking Tool Booking Service Launching Service Experiment Controller Experiment Monitoring Tool	
<b>Validated requirement</b>	PT-GEN-001, PT-P-001, PT-P-003, PT-A-001, PT-A-003, PT-A-004, PT-A-005, PT-A-006, PT-A-008, PT-A-009, PT-A-013, PT-A-014, PT-A-016, PT-B-001, PT-L-002, PT-E-002, PT-E-003	

Step	Description	Status	Remarks
1	Experimenter logs in to the RAWFIE portal with the appropriate credentials		
2	Experimenter looks for the testbeds and UxV resources (simulated resources) available		
3	Experimenter uses the Experiment Authoring tool to write the experiment steps with EDL, e.g. <ul style="list-style-type: none"> <li>o Ask UxV’s current status and location (x1, y1)</li> <li>o Move to location x2, y2</li> <li>o Monitor this location point</li> <li>o Return to the initial location</li> </ul>		
4	Experimenter books the testbed and needed UxVs		
5	Experiment will be started at the given date/time		
6	EDL script is executed correctly using the UxV Generator component as end device that simulates UxVs behavior		
7	Measurements are sent to the database		
8	Experiment finishes		
9	Experimenter evaluates the results <ul style="list-style-type: none"> <li>- View experiment log</li> <li>- Examine measurements</li> </ul>		

Metric	Success criteria	Status	Remarks
PLATFORM / PERF / 3 / SCALABILITY			
PLATFORM / PERF / 5 / LATENCY/ RESULTS UPDATE TIME			



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

PLATFORM / PERF / 6 / LATENCY / BOOKING TIME			
PLATFORM / USE / 7 / NOTIFICATION			
PLATFORM / USE / 8 / ROLES			
PLATFORM / USE / 10 / VISUALISATION / SIMPLICITY			
PLATFORM / USE / 12 / VISUALISATION / UTILITY			
PLATFORM / USE / 13 / GUIDANCE			
PLATFORM / USE / 14 / FILTERING			
TESTBED / USE / 4 / CONSISTENCY			
UxV / FUNC / 1 / COHERENCE			
UxV / FUNC / 2 / MISSION ACHIEVEMENT			
INTERCONNECTIVITY / PERF / 1 / AGGREGATED THROUGHPUT			
INTERCONNECTIVITY / PERF / 1 / COMPONENTS THROUGHPUT			
INTERCONNECTIVITY / PERF / 2 / END-TO-END DELAY			



**7.5.2 UGV navigation and monitoring**

<b>Scenario ID: EST-02</b>	<b>Conducted by:</b>	<b>Date:</b>	
<b>Title</b>	UGV navigation and monitoring		
<b>Comment</b>	A UGV (a ROBOTNIK Summit XL Robot) properly navigates to the coordinates described by end-user experiments and takes some action based on its sensing capabilities (e.g. take photos when predefined coordinates where reached). The scope of this validation scenario is to provide evidence that the UxV node interacts correctly with the RAWFIE platform using the appropriate testbed components and its network communication and navigation subcomponents behave as expected. Besides the Front-end tier, middle tier and data tier this validation test assumes that the Vehicular Testbed (VT) component in the testbed tier is fully functional and running.		
<b>Main stakeholder</b>	Experimenter		
<b>Secondary stakeholder</b>	RAWFIE Platform Administrator / Testbed Operators / Uxv Manufacturers		
<b>Involved Sub-systems</b>	Web Portal Users & Rights Service Resource Explorer Tool Testbeds Directory Service Experiment Authoring Tool EDL Compiler & Validator Experiment Validation Service Booking Tool Booking Service Launching Service Experiment Controller Experiment Monitoring Tool Vehicular Testbed Resource Controller UGV node(s)		
<b>Validated requirement</b>	PT-GEN-001, PT-P-001, PT-P-003, PT-A-001, PT-A-003, PT-A-004, PT-A-005, PT-A-006, PT-A-008, PT-A-009, PT-A-013, PT-A-014, PT-A-016, PT-B-001, PT-L-002, PT-E-002, PT-E-003		
<b>Step</b>	<b>Description</b>	<b>Status</b>	<b>Remarks</b>
1	Experimenter logs in to the RAWFIE portal with the appropriate credentials		
2	Experimenter looks for the testbeds and UxV resources available		
3	Experimenter uses the Experiment Authoring tool to write the experiment steps with EDL, e.g. <ul style="list-style-type: none"> <li>o Ask UGV’s current status and location (x1, y1)</li> <li>o Move to different locations</li> <li>o Monitor these location points</li> <li>o Return to the initial location</li> </ul>		
4	Experimenter books the testbed and needed UxVs		
5	Experiment will be started at the given date/time		
6	EDL script is executed correctly and UGV behaves as expected		
7	Measurements are sent to the database		
8	Experiment finishes		
9	Experimenter evaluates the results <ul style="list-style-type: none"> <li>- View experiment log</li> <li>- Examine measurements</li> </ul>		
<b>Metric</b>	<b>Success criteria</b>	<b>Status</b>	<b>Remarks</b>
All			



## 8 ANNEX 1. Validation scenario template

The templates for the validation scenarios and their evaluation metrics are described in the tables hereafter.

**Table 88: Validation scenario: Scenario 1**

Scenario ID:	Conducted by:	Date:	
<b>Title</b>			
<b>Comment</b>			
<b>Validated requirement</b>			
<b>Technology</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
<b>Measurements</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
<b>Environment</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
<b>Algorithm</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
<b>Special script steps</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
<b>Metric</b>	<b>Success criteria</b>	<b>Status</b>	<b>Remarks</b>



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

Scenario ID:	Conducted by:	Date:	
<b>Title</b>			
<b>Comment</b>			
<b>Validated requirement</b>			
<b>Technology</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
<b>Measurements</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
<b>Environment</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
<b>Algorithm</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
<b>Special script steps</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
<b>Metric</b>	<b>Success criteria</b>	<b>Status</b>	<b>Remarks</b>

Or

Scenario ID: <b>UD-01</b>	Conducted by:	Date:	
<b>Title</b>			
<b>Comment</b>			
<b>Main stakeholder</b>			
<b>Secondary stakeholder</b>			
<b>Involved Sub-systems</b>			
<b>Validated requirement</b>			
<b>Step</b>	<b>Description</b>	<b>Status</b>	<b>Remarks</b>
1			
2			
3			
4			
5			
6			
7			
<b>Metric</b>	<b>Success criteria</b>	<b>Status</b>	<b>Remarks</b>





The validation scenario addressing a specific feature or function of the RAWFIE testbed are described in the tables hereafter.

**Table 89: specific validation scenario: xxxx**

Scenario ID:	Conducted by:	Date:	
<b>Title</b>			
<b>Comment</b>			
<b>Validated requirement</b>			
<b>Technology</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
<b>Measurements</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
<b>Environment</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
<b>Algorithm</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
<b>Special script steps</b>	<b>Details</b>	<b>Status</b>	<b>Remarks</b>
<b>Metric</b>	<b>Success criteria</b>	<b>Status</b>	<b>Remarks</b>

**Table 90: Metrics and success criteria**

<b>Component, feature, function</b>	<b>Short description</b>	<b>Metrics</b>	<b>success criteria</b>	<b>comment</b>
-------------------------------------	--------------------------	----------------	-------------------------	----------------

## 9 ANNEX 2. Component testing - how to read the verification scenarios

Even if at conceptual level in most of the case, the members of the consortium have identified all main system components, both hardware and software, and for each of them the template that will be described in the following will be adopted, in order to describe each required testing scenario.



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

We will assume that each component can consist of zero, one or more sub-components. If there are no sub-components the testing scenario is related to the component itself, whereas in the other cases it will be related to each sub-component.

Moreover, each component (or sub-component) can have one or more verification scenarios.

Two cases can be distinguished:

1. Only one test for the component (or sub-component). In this case the following table template is used:

**Table 91: test for the component (or sub-component)**

Test ID:	Conducted by:	Date:	Test Category: <b>Verification Tests (which tier?)</b>	
<b>Hardware Configuration</b>				
<b>Software Configuration</b>				
<b>Test Name:</b>	<i>Name of the test</i>			
<b>Preconditions</b>	•			
<b>Related Requirements</b>	Requirement IDs from D3.2			
<b>Tools Used</b>				
<b>Step</b>	<b>Action</b>	<b>Expected Result</b>	<b>Status</b>	<b>Remarks</b>
1				
2				
3				
4				

2. More tests for the component (or sub-component). In this case the following template is used:

- A) first template describing the component and identifying the tests that will be performed on that component, by attributing a TEST ID at each test;

**Table 92: tests that will be performed on a given component**

Component (parent component applicable) if	Name of the component
Component Behaviour	If useful, please refer to the corresponding section in D4.1.
Tests	List of the test IDs  All tests in this section...  As in the following the test...



B) a second template describing the specific test and the expected results.

**Table 93: specific test of a given component and the expected results**

Test	Name of the test
Test ID	ID of the test
Component (parent component if applicable)	Name of the component/s involved in the verification test
Pre-requisites	Working condition for the component in order to be able to execute the test
Test description	Condition that should be verified; Sequence of steps to perform the test
Expected results	The expected results from the execution of the steps described.

### 10 ANNEX 3: Analysis of the first Open call

The following table gives a description of the scenarios found in the proposals received in response to the first RAWFIE open call. The relevance of the scenario with respect to validation and testing of the RAWFIE system is evaluated. Then the items that could be validated by the scenario are identified and the feasibility of the scenario is assessed. Finally the decision of making it a standalone scenario or a new step into an existing scenario is mentioned.

**Table 94: Additional scenarios from Open calls**

Description	Relevant (yes/no)	What for?	Is it feasible in RAWFIE?	Standalone scenario or to be combined?
1. Trajectory control during deep-stall: Landing of a fixed wing drone is a complex task that is not completely solved by autopilots nowadays. Factors like turbulence, wind gradient, obstacles nearby and roughness	No			



D4.6 - Pilot Experimentation Scenarios for Validation and Testing

<p>of the landing spot should be considered for a successful and safe landing. In order to avoid all these factors we decided to include the so called “deep stall landing” in our drone. This type of vertical landing has been extensively used by military drones but not for civil ones. As the airplane still moves forward at slow speed when the wing is completely stalled, we need to develop a special aerodynamic configuration that works in deep stall mode and permits the guidance of the drone to the designed landing spot from the place where the deep stall starts.</p>				
<p>2. Navigation, autopilot, communication and obstacle avoidance systems.</p>	<p>Yes</p>	<p>obstacle avoidance</p>	<p>Yes</p>	<p>To be combined in Sc. #4 (see D3.1)</p>
<p>3. Early fire detection, 4. prevention, monitoring, prediction and fire-fighting border UAV surveillance services, 5. maritime border surveillance 6. flight demonstration 7. real-time visual Intelligence and/or mapping on demand capabilities</p>	<p>3 – yes 4 – Yes 5 – Yes 6 – No 7 – yes</p>	<p>3 – see justification of the combined scenario 4 – id 5 – id 6 – N/A 7 – id</p>	<p>All- yes</p>	<p>3+4 combine Sc. #5 5 combined Sc. #2 7 combine Sc. #5, #7 or #8</p>
<p>8. Flight monitoring on the UAV tested into his airspaces, beyond peer to peer and</p>	<p>Yes</p>	<p>Possibility to compare different technologies (e.g. accuracy of the</p>	<p>Yes</p>	<p>Standalone</p>



D4.6 - Pilot Experimentation Scenarios for Validation and Testing

standard GNSS quality (embed several communication protocols and frequencies into the tracking system plus a selective accurate GNSS components).		positioning)		
9. The geofencing service will be improved, providing a better security for the operations as well as a good quality of the data for the flights logs analysis by the experimenters. The SES component will be evaluated related to the geofencing service performance. Experimenters using CESA-Drones sites will have the possibility to perform various tests such as “long distance” scenario (up to 50 km in a segregated area) Experimenters will access to high end tools such as performant and accurate geofencing	Yes, to be reformulated, details to be precised	Safety security of the platform (ethics)	TBC	Depends on the definition of the scenario, but probably standalone
10. Acoustic localization and communication link for an underwater agent (vehicles, divers)	No			
11. Deployment of mesh networks in emergency contexts for purposes of providing communications and connectivity, 12. carrying of small items (for instance, medical equipment),	11 – yes 12 - no 13 – yes, already considered	Provide Internet access in remote areas	11 – yes 12- N/A 13 - yes	11 – standalone (see also Sc. #3) 12 – N/A 13 – Sc. #1, sc. #5 and sc. #8



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

13. monitoring and reporting on a variety of parameters, ranging from weather conditions and environmental measurements to detection of movement/intrusions and more.				
14. Inspection, monitoring and surveillance capabilities can also find useful applications in contexts of large, not easily accessible infrastructures such as in the construction and agricultural industry, in cases of contexts that lack communications (such as in maritime areas) or in remote or hazardous for humans locations.	Yes	See the justifications of the scenario in which it is merged	Yes	Sc. #2, Sc. #5, Sc. #7
15. Large scale imaging and 3D visualization or aerial photography	Yes	Long-term flights, high data throughput, precise GNSS, collaboration between UxVs for efficiency	Yes (provided we have the sensors)	Standalone
16. Special police or anti-terrorist indoor interventions, with an aerial overview for identification and localization and threat localization purposes,	Yes	Indoor (localization, communication issues, etc.)	Yes (e.g. Saragossa)	Standalone
17. Indoor safety inspection with UGV or UAV,	Yes	Id	Yes	see above, combine them
18. Industrial inspection with a USV UAV or UGV robots USV (thick walls and their inherent communication	Yes	(no GPS, use relay, full or partial autonomy)	Yes	see above, combine them



issues),				
19. Indoor Fire search and rescue operations: nodes, relays, communications using UxVs,	Yes (e.g. earthquake, emergency SAR...)	for trying high-level control	Yes	See above
20. Simulated industrial and safety inspection in large water pipes and liquid environments with small Underwater unmanned vehicles: nuclear plants, pipeline inspection, etc.	Yes	Pipes: specific location/environment. It will be simulated in a pool	Yes	Extend Sc; #1

## 11 ANNEX: Unreferenced Requirements

This table provides an overview of the unreferenced requirements of D3.2.

Table 95 – Unreferenced Requirements

No	ID	Component	Category	Title	Type	Priority
1	PT-GEN-R-003	General	PLATFORM	The RAWFIE Data model should include all basic entities that are used or/and exchanged by the various components of the RAWFIE Platform	DATA	<b>HIGH</b>
2	PT-BOO-T-013	Booking Tool	PLATFORM	RAWFIE platform should allow virtualization of available UxVs resources during reservation process	FUNC	<b>LOW</b>
3	PT-SYM-T-002	System Monitoring Tool	PLATFORM	The current system health status should be grouped thematically.	FUNC	<b>MEDIUM</b>
4	PT-SYM-T-005	System Monitoring Tool	PLATFORM	The health status information should include a severity indication and possibly textual information with additional details.	FUNC	<b>HIGH</b>
5	PT-REE-T-002	Resource Explorer Tool	PLATFORM	Registration of testbeds and UxVs may be possible via the Web Portal	FUNC	<b>LOW</b>
6	PT-EXA-T-007	Experiment Authoring Tool	PLATFORM	An experimenter shall be able to define the type of	FUNC	<b>HIGH</b>



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

				metrics to be gathered and/or stored during an experiment and/or per UxV resource		
7	PT-EXA-T-014	Experiment Authoring Tool	PLATFORM	During authoring of an experiment selection of resources should be limited only to the ones previously reserved from the user at the foreseen time of experiment	FUNC	<b>HIGH</b>
8	PT-EXA-T-016	Experiment Authoring Tool	PLATFORM	An experimenter shall have the means to define actions or tasks that should run on a periodic or ad hoc basis during execution of an experiment	FUNC	<b>MEDIUM</b>
9	PT-NAV-T-003	UxV Navigation Tool	PLATFORM	UxV Navigation Tool should be available for the navigation of all moving resources	FUNC	<b>HIGH</b>
10	PT-NAV-T-004	UxV Navigation Tool	PLATFORM	UxV Navigation Tool should be available to read from the database a detailed version of the map of the available areas	FUNC	<b>HIGH</b>
11	PT-VIS-T-001	Visualisation Tool	PLATFORM	The Visualisation Tool shall allow the visualisation of information about the running experiments, in tabular/graphical form	FUNC	<b>HIGH</b>
12	PT-VIS-T-003	Visualisation Tool	PLATFORM	The Visualisation Tool may allow visualisation of video streams coming from the experiment, and experiment's camera control	FUNC	<b>LOW</b>
13	PT-DAA-T-001	Data Analysis Tool	PLATFORM	Analysis tool will provide interface to data engine.	FUNC	<b>MEDIUM</b>
14	PT-DAA-T-002	Data Analysis Tool	PLATFORM	Analysis tool will provide access to past experiments	FUNC	<b>LOW</b>
15	PT-DAA-T-003	Data Analysis Tool	PLATFORM	Analysis tool will provide ability to query message bus streams	FUNC	<b>MEDIUM</b>
16	PT-DAA-T-004	Data Analysis Tool	PLATFORM	Analysis tool will provide interface to end running jobs	FUNC	<b>MEDIUM</b>





## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

17	PT-DAA-T-005	Data Analysis Tool	PLATFORM	Analysis tool will provide a simple metric selection interface, a view of the result stream & the job status tab		
18	PT-DIR-S-005	Testbeds Directory Service	PLATFORM	The Testbed Directory Service should provide the possibility to register new testbeds in the RAWFIE platform, as well as to unregister (delete) testbeds from the platform	FUNC	<b>HIGH</b>
19	PT-DIR-S-007	Testbeds Directory Service	PLATFORM	The Testbed Directory Service shall provide the possibility to register new resources belonging to a specific testbed in the RAWFIE platform, as well as to unregister (delete) resources	FUNC	<b>HIGH</b>
20	PT-LAU-S-011	Launching Service	PLATFORM	RAWFIE platform shall provide means to ensure fairness in experiments execution	FUNC	<b>MEDIUM</b>
21	PT-VIS-E-003	Visualisation Engine	PLATFORM	The Visualization Engine may allow cache of data for faster access to the available geographic layers	FUNC	<b>MEDIUM</b>
22	PT-SYM-S-005	System Monitoring Service	PLATFORM	Notifications about planned downtimes	FUNC	<b>MEDIUM</b>
23	PT-ACC-S-001	Accounting Service	PLATFORM	The accounting service should be capable to accept different cost models regarding RAWFIE usage on a per service basis	FUNC	<b>MEDIUM</b>
24	PT-ACC-S-004	Accounting Service	PLATFORM	The cost model used may take into consideration the overall time of experiments executed by a user of the platform.	FUNC	<b>MEDIUM</b>
25	PT-ACC-S-005	Accounting Service	PLATFORM	The accounting service may support different types of charging based on the type of the experimenter (industrial, research, university etc.)	FUNC	<b>MEDIUM</b>
26	PT-ACC-S-	Accounting	PLATFORM	The accounting service	FUNC	<b>MEDIUM</b>



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

	006	Service		may support predefined types of memberships regarding usage of the platform that may depend on various types of parameters		
27	PT-ACC-S-007	Accounting Service	PLATFORM	The accounting service should be able to handle the addition of new services that may be incorporated in the RAWFIE platform during time.	FUNC	<b>MEDIUM</b>
28	TB-GEN-R-003	General	TESTBED	Testbed areas should at least be able to host/operate multiple UxVs of one or more types	FUNC	<b>HIGH</b>
29	TB-GEN-R-004	General	TESTBED	Testbed areas environment should be closely monitored	ENV	<b>HIGH</b>
30	TB-GEN-R-005	General	TESTBED	Indoor spaces of a testbed should provide a controlled indoor environment	ENV	<b>HIGH</b>
31	TB-GEN-R-006	General	TESTBED	Testbed facility areas should comprise storing spaces and be able to receive inspect and assemble and/or fix UxVs	SUPP	<b>HIGH</b>
32	TB-GEN-R-007	General	TESTBED	Testbed facilities should provide emergency services in an extraordinary event	SEC	<b>HIGH</b>
33	TB-GEN-R-008	General	TESTBED	Testbed areas should provide proper facilities and equipment	ENV	<b>HIGH</b>
34	TB-GEN-R-009	General	TESTBED	Testbed must provide dedicated computational resources	ENV	<b>HIGH</b>
35	TB-GEN-R-010	General	TESTBED	Testbeds should be supported by on-site personnel	OTH	<b>HIGH</b>
36	TB-GEN-R-011	General	TESTBED	Testbeds should conform to all legal regulations and restrictions	SEC	<b>HIGH</b>
37	TB-NEC-002	Network Controller	TESTBED	Provision of network communication resource	FUNC	<b>MEDIUM</b>
38	TB-NEC-005	Network Controller	TESTBED	Time constraint verification and notification	FUNC	<b>MEDIUM</b>



## D4.6 - Pilot Experimentation Scenarios for Validation and Testing

39	<b>TB-REC-002</b>	Resource Controller	TESTBED	RAWFIE platform should be able to activate the "Emergency Scenario"	FUNC	<b>MEDIUM</b>
40	TB-REC-003	Resource Controller	TESTBED	The Resource Controller shall receive location messages from the vehicles at regular intervals	FUNC	<b>HIGH</b>
41	TB-REC-006	Resource Controller	TESTBED	For the experiment accomplishment the Resource Controller shall operate in close coordination with the Experiment Controller	FUNC	<b>HIGH</b>
42	TB-UVG-001	General	UxV	Compliance of UxV to RAWFIE specification and interfaces	FUNC	<b>HIGH</b>
43	UXV-NOD-001	UxV Node	UxV	Each UxV shall have a unique Identification code.	FUNC	<b>HIGH</b>
44	UXV-NOD-003	UxV Node	UxV	Each UxV node should ensure payload.	FUNC	<b>HIGH</b>
45	UXV-SEN-004	UxV Sensor and Localisation	UxV	Location sensors should be supported in each UxV unit and can be used remotely during testbed demonstrations.	FUNC	<b>HIGH</b>
46	UXV-STO-005	UxV On-board storage	UxV	UxV's may provide an automated syncing of servers.	FUNC	<b>MEDIUM</b>
47	UXV-PRC-003	UxV On-board processing	UxV	Capability of task planning of the UxVs nodes during run-time.	FUNC	<b>MEDIUM</b>
48	UXV-PRC-005	UxV On-board processing	UxV	Each UxV node shall keep position while waiting for new instructions.	FUNC	<b>HIGH</b>
49	UXV-MGT-001	UxV Management	UxV	UxVs shall offer on demand resources (Network, Sensor, Processing, and Controller).	OTH	<b>HIGH</b>
50	UXV-MGT-003	UxV Management	UxV	UxV shall be capable to restart its internal components independently	FUNC	<b>HIGH</b>
51	UXV-MGT-006	UxV Management	UxV	UxV shall be capable to offer safe maintenance access for manufacturers	OTH	<b>HIGH</b>



## 12 References

- [FFF D6.1] Detailed specifications for first cycle ready, Fed4FIRE D6.1 deliverable, 2013.  
[http://www.fed4fire.eu/fileadmin/documents/public\\_deliverables/D6-1\\_Fed4FIRE\\_Detailed\\_specifications\\_for\\_first\\_cycle\\_ready.pdf](http://www.fed4fire.eu/fileadmin/documents/public_deliverables/D6-1_Fed4FIRE_Detailed_specifications_for_first_cycle_ready.pdf)
- [RAWFIE D3.2] Specifications and Analysis of RAWFIE Components Requirements, RAWFIE D3.2 deliverable, 2016.
- [RAWFIE D4.5] Design and Specifications of RAWFIE Components, RAWFIE D4.5 deliverable, 2016.